PhD Thesis

DIETI, University Federico II of Naples

Dottorato in Information Technology and Elettrical Engineering

XXX Ciclo

Source identification in image forensics

Francesco Marra

Tutor
Prof. Carlo Sansone

Co-Tutor
Prof. Luisa Verdoliva

Contents

List of Tables

List of Figures

Introduction

Source identification is one of the most important tasks in digital image forensics. In fact, the ability to reliably associate an image with its acquisition device may be crucial both during investigations and before a court of law. For example, one may be interested in proving that a certain photo was taken by his/her camera, in order to claim intellectual property. On the contrary, it may be law enforcement agencies that are interested to trace back the origin of some images, because they violate the law themselves (e.g. do not respect privacy laws), or maybe they point to subjects involved in unlawful and dangerous activities (like terrorism, pedo-pornography, etc). More in general, proving, beyond reasonable doubts, that a photo was taken by a given camera, may be an important element for decisions in court.

The growing interest towards camera identification is also a consequence, on the other hand, of the capillary diffusion of imaging devices, and of the widespread diffusion of images on the net. It is estimated that in 2014 more than 1.8 billion images and videos have been published each day, and this trend does not seem to be slowing down. The analyst may seek information at various levels, from the type of source (camera, scanner, etc.), to its brand/model (e.g. iPhone6 vs iPhone7), to the individual

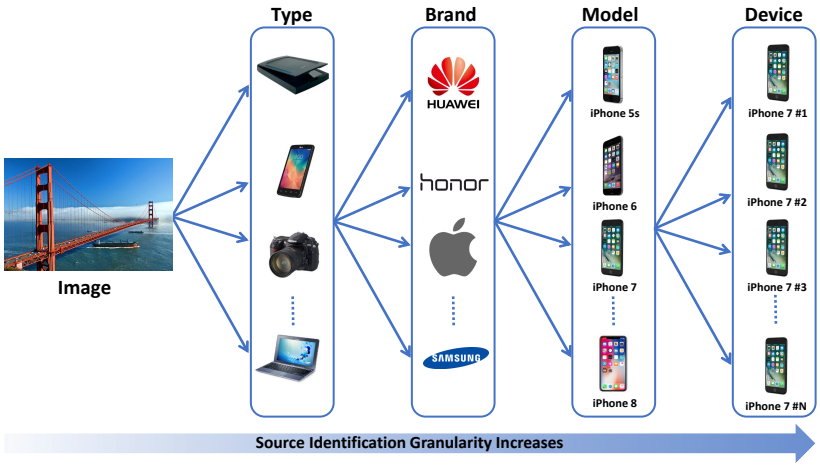


Figure 1. Granularity levels in forensic source identification.

device, as shown in Fig.1 [1]. The identification of the specific acquisition device is often desired, but not always possible and in that case it makes sense to work at a higher level of granularity. A possible approach to recover the source device information of a media is to look at the meta-data headers itself, e.g. the exchangeable image file format (EXIF) of a photograph. Unfortunately, these headers can be easily removed or counterfeited even by a beginner forger using simple editing tools.

The key assumption of forensic source identification is that acquisition devices leave traces in the acquired content, and that instances of these traces are specific to the respective (class of) device(s). This kind of traces is present in the so-called *device fingerprint* (Fig.2). The name stems from the forensic value of human fingerprints. In the ideal case, the device fingerprint has two properties: diversity and stability. The diversity requires that it is unique and not shared among different camera models or devices, while stability requires that the fingerprint remains the same over time.

A major impulse to research in this field came with the seminal work of Lukas *et al.* [2] showing that reliable device identification is possible

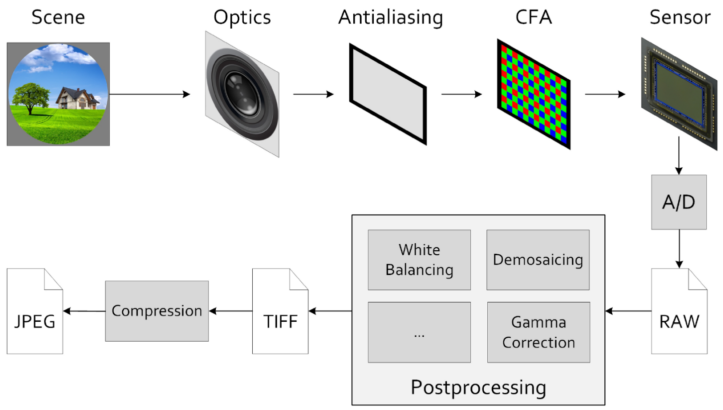


Figure 2. The digital image acquisition chain left a so-called *device fingerprint* on each acquired image.

based on the camera photo-response non uniformity (PRNU) pattern. This is a multiplicative noise component caused by the inhomogeneity of silicon wafers and imperfections of the sensor manufacturing, which, in turn, cause a non-uniform sensitivity to light among the sensor photo-diodes. This means that a pixel could be slightly brighter or darker than expected by camera design, and each pixel is individually affected by this issue. Each camera is characterized by its unique PRNU pattern, which can be regarded as a sort of *camera fingerprint*. All photos taken by a given camera carry traces of its fingerprint which, under suitable hypotheses, can be retrieved, enabling reliable device identification and, with some further processing, also brand and model identification [3]. Camera brand and model identification are based also on the other traces left by the internal processing steps, like the lens aberration, the demosaicing algorithm, the CFA and the compression matrix. Of course, the *device fingerprint* itself must be known in advance, or estimated from a large set of photos taken by the desired source, a restrictive hypothesis that limits somewhat the applicability of this approach.

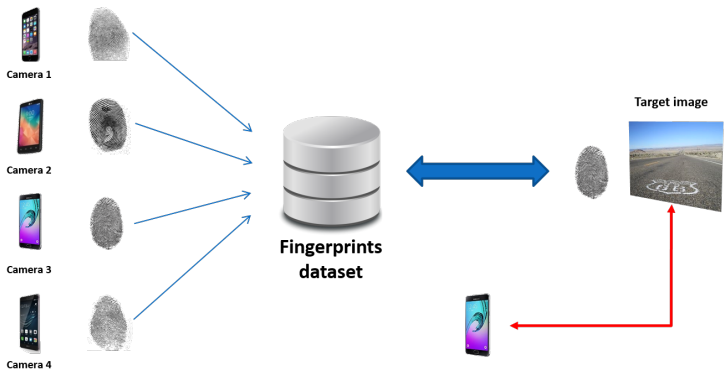


Figure 3. Source identification performed in a *perfect knowledge* scenario. In this scenario the fingerprint of the image under test is compared to a known fingerprints extracted from a reference labelled data.

The device identification problem takes different forms depending on the prior knowledge available. If the target image may only come from one of a given number of cameras, whose *device fingerprint* is known in advance or can be accurately estimated, identification reduces to a classification problem [2, 4, 5]. This is called *perfect knowledge* scenario (Fig.3). In a *limited knowledge* situation with a more challenging open set scenario, the target image may also come from unknown sources [6] and the problem is to understand whether it was acquired by one of the known cameras (possibly just one) or not [7]. Often, however, the analyst has only a set of images without any information on the possible device involved [8]. In this *zero knowledge* scenario the only approach is to perform a *blind* clustering of the images respect to their source (Fig.4).

Camera model identification and PRNU-based estimation can be also used for image forgery detection. This is also a very hot topic in these recent years, given the availability of modern and powerful image editing software tools. Almost everyone can produce dozens of new digital pictures each day and share them through social networks. For the large majority,

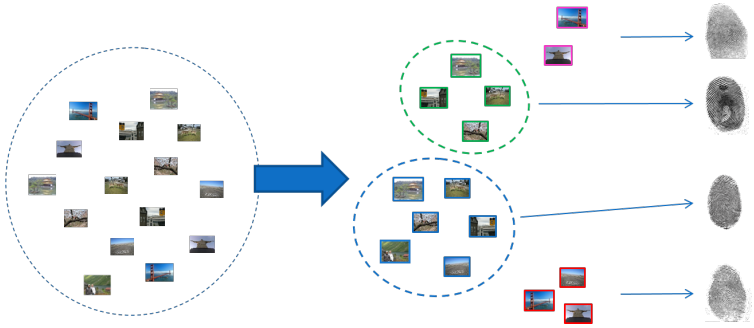


Figure 4. Source identification performed in a *blind* scenario. Here we have no labelled data, and we want to extract from the single-image fingerprints as much information as possible. For example here, we cluster the data respect to the device that took the images.

they are genuine images, however they are often manipulated and diffused with malicious purposes, like influencing the public opinion or discrediting people. Such attacks are becoming more and more frequent and sophisticated, raising a serious alarm over the general trustfulness of multimedia assets. Many techniques, especially machine-learning ones, have been proven to be a successful approach to deal with this task [9]. However, an expert adversary could be able to fool them. Counter-forensics is the research field that studies forensic techniques, finds their weak points, and tries to exploit them to fool forensic approaches. It is therefore important to discover even the weakness of forensics tools in order to take specific actions and propose ever more robust techniques.

Motivated by the importance of the source identification in *digital image forensics* community and the need of reliable techniques using device fingerprint, the work developed in this PhD thesis concerns different source identification level, using both feature-based and PRNU-based approach for model and device identification. In addition, it is also shown that counter-forensics methods can easily attack machine learning techniques for image forgery detection. The thesis is hence organized so as to devote

a chapter to these different problems. In more detail,

Chapter 1 deals with feature-based techniques for camera model identification. Specifically, an analysis of hand-crafted local features and deep learning ones will be considered for the basic two-class classification problem. In addition, a comparison with the limited knowledge and the blind scenario will be presented. Finally, an application of camera model identification on various iris sensor models will be shown.

Chapter 2 faces the problem of device source identification using the PRNU-based approach in a blind scenario. With the use of the correlation between single-image sensor noise, a blind two-step source clustering is proposed. In the first step correlation clustering together with ensemble method is used to obtain an initial partition, which is then refined in the second step by means of a Bayesian approach. Experimental results show that this proposal outperforms the state-of-the-art techniques and still give acceptable performance when considering images downloaded from Facebook.

Chapter 3 presents an application of forgery localization in a blind scenario. The source image clustering algorithm presented in chapter 3 is used in a realistic scenario to carry out image forgery detection when PRNU is not available.

In **Chapter 4** it is presented a counter-forensics technique based on a greedy strategy that attacks a machine-learning based method using handcrafted features. The analysis is carried out on a synthetic datasets of forged images and shows that a malicious attacker can easily fool the detector using the proposed approach.

Chapter 1

Learning-based source identification

Camera model identification relies on the distinctive traces left in images by the processing steps carried out inside modern cameras. In fact, in digital cameras, the output image is obtained by applying a number of sophisticated algorithms, each characterized by several free parameters. Well-known examples are demosaicing, based often on complex adaptive nonlinear interpolation, and JPEG compression, where the quantization matrix can be defined by the user. Each camera model is characterized by its own combination of in-camera algorithms. It is highly unlikely that different camera models, even of the same brand, use the very same set of algorithms and parameters and, therefore, very likely that their traces allow reliable identification.

In literature there are a number of papers that try to solve the problem looking for artifacts related to *specific* in-camera processing steps, trying to estimate their unknown parameters. However, a *blind* approach is also possible, where no hypothesis is made on the origin of camera-specific

marks, and the identification task is regarded simply as a texture classification problem. With this approach, the focus shifts on the definition of the most discriminative features, irrespective of their physical meaning. Both global and local features can be considered, drawing often from the vast literature of closely related fields, such as material classification or steganalysis. Recently, the use of deep learning and CNNs permits to extract such feature directly learning from the data the useful feature that permit to distinguish between camera models. This so-called *data driven* approach needs a big labeled dataset for training but guarantee a big improvement on performance.

The aim of this chapter is to evaluate a class of such feature on different scenarios. First a *perfect knowledge* scenario is considered where we have information of the camera model source of the images in our dataset. Then a *limited knowledge* and *zero knowledge* scenarios are taken in account.

In addition both a *blind* and *data driven* feature are proposed. The first uses the co-occurrences of image residuals [10] assessing their potential for the camera model identification task. It is worth noting that co-occurrence based local features have been also applied recently [11], together with some features proposed in [3]. In [12] we have tested the very same approach, starting from a state-of-the-art denoising filter [13], and found it to be inferior to that based on simple high-pass filtering. In this preliminary results only grayscale images were used, and only a perfect-knowledge scenario was considered.

A *data driven* based on CNN is also proposed. Respect to other CNN-based approaches recently proposed [14, 15], the goal is to keep the net relatively small, to be used on small datasets and to keep the complexity quite low. Since for training they need a full labeled dataset, only a *perfect knowledge* scenario is considered.

In the next Section we review model identification methods based on both “in-camera specific”, “blind” and “data-driven” features, then, in Sec-

tion 1.2, we describe the co-occurrence based one proposed in [16] and the CNN used in [17]. Experimental analysis on the Dresden Image Database, is carried out in Section 1.4 considering various scenarios of interest. Finally, in Section 1.5 we show the application of model identification in the context of as iris sensor model recognition.

1.1 Related work

For solving the problem of camera model identification, two different approaches were used. The first try to exploit the artifacts related to specific in-camera processing steps, using *hand-crafted features*. The second approach uses *blind-features*, sometimes borrowed by other fields, for solving the camera model identification task.

Model-based artifacts were recognized also in early research on PRNU-based identification [3], during the PRNU pattern estimation, and used to identify the camera model. This path has been followed later also by other researchers with different type of features [11, 18]. In [19], inspired by the work of Popescu and Farid on image forgery detection [20], traces of different interpolation algorithms where sought and used as distinctive model features. In fact, interpolation algorithms modify in specific ways, both in space and across color channels, the natural correlation between each pixel and its neighbors. Therefore, the weights of the interpolation kernel, once estimated, can be used as features for camera identification. Often, they are combined with frequency domain features, that take into account the periodic artifacts caused by the color filter array (CFA) pattern. The strong dependencies among pixels has been also explored in [21] and [22]. In [21], in particular, as also in [23], weight estimation is conducted locally on each color band using a content-adaptive procedure for each region. This reflects the fact that, often, adaptive demosaicing techniques are used inside the camera to reduce blurring artifacts. In [22],

instead, partial second-order derivative correlation models are proposed to detect both the intra-channel and the cross-channel dependencies due to demosaicing. Other methods aim at characterizing JPEG compression artifacts [24], DCT coefficients statistics [25], or lens distortion artifacts like chromatic aberration [26]. A different approach is proposed in [27] where identification is based on a two-parameter heteroscedastic noise model valid for raw images.

Kharrazi et al. [28] in 2004 considered the use of generic features for camera identification. This was one of the first papers to present an approach that did not focus on a specific camera artifact, but tried to capture the underlying variations between camera models based on statistics of various orders. The authors propose to use several global statistics, extracted from each individual color band, based on the correlation of couples of color bands, and also extracted from some wavelet subbands. In addition, some Image Quality Metrics (IQM), previously used in [29] for steganalysis, are evaluated on all the color bands, both in the spatial and transform domain. It is important to underline that these last features are computed on residual images (high-pass filtered versions of the original data). These features have shown good performance also for cell phone identification tested with both images and videos [30].

Image residuals are used by the majority of methods proposed in the literature. The reason is that in this way results become independent of the image content, hence artifacts are more easily detected. In [31] IQM features are extracted from high-pass residuals of each color band. These features are then combined with BSM (Binary Similarity Measures), i.e., LBP (Local Binary Pattern) [32] extracted from the least-significant bit planes, and with an enlarged set of features computed in the wavelet domain. Besides the features used in [28], other first-order statistics are computed, as well as some inter-band correlation indexes inspired by [33]. In order to improve performance Gloe [34] proposes to add some color

features to those used in [28]. Effectively, experiments on the Dresden Image Database prove this combination to guarantee a performance gain w.r.t. both [28] and [31].

Most of the above described features are evaluated globally on the whole image (both original and high-pass filtered) or on a decimated version of it, if wavelet subbands are considered. However, in order to capture subtle image patterns which may correspond to discriminative features, it is important to consider local features, extracted from a small neighborhood of each pixel of the image. This idea inspires the work in [35] where LBP features are evaluated both on the original image and on some residuals. A similar approach is followed recently also in [36]. Note that in [35] LBP is computed on two-pixel supports, and hence encodes only first-order spatial variations. Computing it after a preliminary high-pass filtering, as done in [37], is instead equivalent to use a larger support and evaluate higher-order statistics. A different approach looks for the statistical differences in the DCT domain by computing Markovian transition probabilities [38].

Recently, the promising performance obtained in various computer vision tasks with the use of deep learning inspired various work on model identification using CNNs [14, 15]. In [15], using a deep learning approach, the authors train a multi-class CNN for extracting discriminant feature for model identification followed by an SVM trained using patches extracted from training images.

1.2 Proposed approaches

In this section we explain in detail the two kind of approaches proposed for camera model identification. A *blind* local feature, based on co-occurrence of image residuals, are first described. Then a *data-driven* and deep learning approaches using CNN is presented.

1.3 Co-occurrence based local features

The analysis of the state of the art shows that local descriptors can provide precious clues for camera model identification. Moreover, since such clues, related to the camera processing chain, are contained in the image micro-patterns and not in the scene content, it makes sense to remove the latter and work on image residuals. Even in this framework, however, two main open issues remain about *i)* how to extract informative image residuals and *ii)* how to process them in order to obtain an expressive camera-related feature. Given the complexity and the variety of the in-camera processes involved, no conclusive answer can be hoped for. However, we will show that co-occurrence based local features, computed on image residuals, and originally proposed in [10] for steganalysis, may represent a valuable tool for this task. A similar path was successfully used in digital image forensics [39] [9] [40].

The feature vector associated with the image under test is extracted by means of the following steps:

- computation of residuals through high-pass filtering;
- quantization and truncation of the residuals;
- computation of the histogram of co-occurrences.

In [10] a number of linear and non-linear high-pass filters have been used for the computation of residuals, and all resulting feature vectors have been combined by means of an ensemble classifier. In [12], instead, inspired also by [40], only a few filters have been selected for the model identification purpose, after a preliminary performance analysis on the training set. In both cases, the input was a gray-scale image, obtained by suitably combining the three color components. Individual color channels, however, are involved in all in-camera processes, and may contribute more

information than their combination. Hence, it makes sense computing co-occurrences based on these richer sources, provided that enough data are available to carry out reliable estimates.

With this aim, we consider the color-aware features proposed in [41] and [42]. In both cases, image residuals are computed separately for each color channel, while differences arise concerning *which* co-occurrences are taken into account. In [41] each color band is processed individually, hence only spatial co-occurrences are considered. Several co-occurrence matrices are computed, one for each color channel and each spatial direction, and they are eventually merged in a single feature vector. In [42], instead, only inter-channel co-occurrences are computed, taking also into account the Bayer CFA configuration of the image. Therefore, we considered both approaches, and performed some preliminary tests in order to select the best solution. It turned out that, for the model identification problem, processing the color channels independently from one another provides better and more stable results, so we consider the approach of [41] in the following.

Filter names are built as in [10] to reflect their main characteristics, that is

$$\text{name} = \text{s}\{\text{order}\}\_\{\text{type}\}\{\text{f}\}\{\sigma\}\{\text{scan}\} \tag{1.1}$$

where order is the filter order, the type can be linear, called spam, or minmax, with the latter case meaning the the output of multiple filters are combined through nonlinear min and max operations, f is the number of filters used, σ a symmetry index (indicating the number of different residuals that can be obtained by image rotation or mirroring), and scan may be h (horizontal), v (vertical) or missing (this accounts for hv-symmetrical residuals). As already said, after some experiments, we focused on a small number of filters, i.e., linear filters of the second and third order and a minmax filter based on the output of multiple second order filters. Let us

focus for the moment on the second order filter, s2_spam12hv, defined by

$$r^h_{i,j} = x_{i,j-1} - 2\,x_{i,j} + x_{i,j+1} \tag{1.2}$$

where x represents a generic color band of the input image (red, green or blue), and r^h the corresponding residual image, with the superscript h indicating the scanning direction. A similar definition applies, with obvious changes, for $r^v_{i,j}$. However, by symmetry, r^h and the transpose or r^v have the same statistics, so they are concatenated in a single residual image r, thus augmenting the data for co-occurrence computation. In order to obtain manageable co-occurrence matrices, residuals are quantized/truncated to a small number of values as:

$$\hat{r}_{i,j} = \mathrm{trunc}_T(\mathrm{round}(r_{i,j}/q)) \tag{1.3}$$

with q the quantization step and T the truncation value, which in this work are set to $q = 1$ and $T = 2$, respectively. The co-occurrences are computed on four pixels in a row aligned along the horizontal (along filter) and vertical (cross filter) directions

$$C^h(k_0,k_1,k_2,k_3) = \sum_{i,j} I(\hat{r}_{i,j} = k_0, \hat{r}_{i,j+1} = k_1, \hat{r}_{i,j+2} = k_2, \hat{r}_{i,j+3} = k_3) \tag{1.4}$$

$$C^v(k_0,k_1,k_2,k_3) = \sum_{i,j} I(\hat{r}_{i,j} = k_0, \hat{r}_{i+1,j} = k_1, \hat{r}_{i+2,j} = k_2, \hat{r}_{i+3,j} = k_3) \tag{1.5}$$

where $I(A)$ is the indicator function of event A, equal to 1 if A holds and 0 otherwise.

With the selected parameters, each of these matrices have 625 entries, which are reduced to 313 by symmetry. Considering the three color channels, the f a fully manageabinal s2_spam12hv feature vector obtained

through concatenation has 3×2×313=1878 components,le length for the typical size of modern images.

The same processing steps are used to extract features based on other filters, with only minor changes. Actually, when the third order filter, s3_spam14hv, is considered, defined as

$$r^h_{i,j} = x_{i,j-1} - 3\,x_{i,j} + 3\,x_{i,j+1} - x_{i,j+2} \tag{1.6}$$

no modification to the process is necessary, and the only observable difference will be in the eventual performance.

On the contrary, the third features we consider, s2_minmax32, require some further considerations. The min residual image $r^{\min}$ is computed by taking, at each pixel, the minimum output of four linear filters (of the second order in our case) operating along rows, columns and the two diagonals (main, *md*, and anti, *ad*)

$$r^{\min}_{i,j} = \min(r^h_{i,j}, r^v_{i,j}, r^{md}_{i,j}, r^{ad}_{i,j}) \tag{1.7}$$

A similar definition applies to $r^{\max}$, where the maximum instead of the minimum is taken. Due to directional symmetry, only two co-occurrence matrices need be computed, one for $r^{\min}$ and one for $r^{\max}$, which are eventually merged. After concatenating the three color-band matrices, the final feature vector has 3×625=1875 components. In all cases, the extracted features are eventually used to train an SVM linear classifier.

1.3.1 CNN for model identification

The architecture of the network proposed for the identification is described in detail in Fig. 1.1. It is an adapted and reduced version of the AlexNet [43]. Moreover, for keeping low the number of parameters, the computational complexity (both in training and testing phases) and the

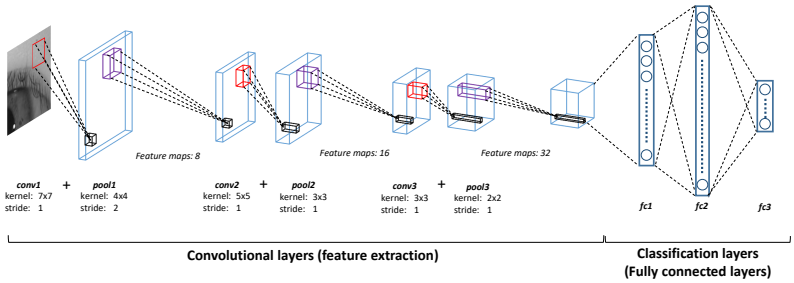


Figure 1.1. The proposed CNN architecture

memory required. It is made up of three convolutional layers for feature extraction followed by three fully connected layers for the classification.

All the convolutional layers are made up of a convolution, followed by the activation function and a pooling operation. We use the Rectified Linear Units (ReLUs) as non-linearity activation function and the *max* function as pooling. The chosen convolutional layers has a decreasing kernel size for both the convolutional and pooling, while an increasing feature map size. The first layer is made up of a convolution with a kernel $7\times$ 7 and a pooling of 4×4, that produces 8 feature maps of the selected patch of the original iris image. The second layer is composed by a convolution with a kernel of 5×5 and a pooling of a kernel 3×3 that produces 16 feature maps. Finally, the third layer with a convolution with a kernel $3\times$ 3 and a pooling of $2\times$ 2 produces the final image representation of 32 features map.

The fully-connected layers *fc1* and *fc2* have 1024 and 2048 neurons respectively and ReLUs as activation function. The last fully connected layer (*fc3*) has the same number of neurons of the available models in the dataset, and, finally, a softmax function is used for classification.

1.4 Experimental results

In order to assess the performance of the described feature for camera model identification, we carried out a number of experiments on two different kind of images: natural images and biometrics sensor one. For the first, we will use the well-known Dresden Image Database [44]. In the first part of the experimental analysis we show the comparison of various blind feature-based proposed for camera identification analysing the performance at various level of knowledge. Then, we show the performance gain given by the use of CNN on model identification. For the biometrics sensor images, we will use a collection of Iris sensor images. In the analysis, we will show that the features used for natural images perform very good on biometrics sensor, and will use a CNN-based approach proposed by us in [17] for iris sensor.

1.4.1 Datasets

1.4.1.1 Natural images

The Dresden Dataset is one of the most widespread database in the forensics field, used in many recent papers. As shown in Table 1.1, 26 different camera models are available, often with several individual devices, and several hundred photos. A limited set of scenes is selected, and portrayed over and over across different models, devices and settings. This is unique characteristic of the Dresden Image Database, which makes it especially suitable for the model identification problem, as it frees experimental results from the randomness due to varying image content.

1.4.1.2 Iris images

For evaluating the performance, we consider some publicly available iris databases. Since there is not one single database explicitly made for

Make	Model	#Devices	Images size	#images
Agfa	DC-504	1	4032 x 3024	169
Agfa	DC-733s	1	3072 x 2304	278
Agfa	DC-830i	1	3264 x 2448	363
Agfa	Sensor505-X	1	2592 x 1944	172
Agfa	Sensor530s	1	4032 x 3024	372
Canon	Ixus 55	1	2592 x 1944	224
Canon	Ixus 70	3	3072 x 2304	567
Canon	PowerShot A640	1	3648 x 2736	188
Casio	EX-Z150	5	3264 x 2448	924
Kodak	M1063	5	3664 x 2748	2391
Nikon	CoolPix S710	5	4352 x 3264	925
Nikon	D200	2	3872 x 2592	752
Nikon	D70/D70s	2/2	3008 x 2000	736
Olympus	μ1050SW	5	3648 x 2736	1040
Panasonic	DMC-FZ50	3	3648 x 2736	931
Pentax	Optio A40	4	4000 x 3000	638
Pentax	Optio W60	1	3648 x 2736	192
Pratika	DCZ5.9	5	2560 x 1920	1019
Ricoh	GX100	5	3648 x 2736	854
Rollei	RCP-7325XS	3	3072 x 2304	589
Samsung	L74wide	3	3072 x 2304	686
Samsung	NV15	3	3648 x 2736	645
Sony	DSC-H50	2	3456 x 2592	541
Sony	DSC-T77	4	3648 x 2736	725
Sony	DSC-W170	2	3648 x 2736	405
Σ	25	74		16956

Table 1.1. Digital camera models used in our experiment, all cameras are taken from Dresden Image Database.

the model identification task, with various models and vendors, we built a dataset by merging all or parts of the model from each selected database.

In Table 1.2, we report the nine models used and the four original databases from which they come, as well as the number of images and their size. In Figure 4, we show some images coming from different sensor models. In the following we recall some of the characteristics of the four considered databases.

Model	Original Database	# Images	Im. size
LG Iris Access EOU3000	ATVS-FIr_DB [1]	1600	640 x 480
CASIA-IrisCamV2	CASIA-IrisV2 [2]	1200	640 x 480
OKI IRISPASS-h	CASIA-IrisV2 [2]	1200	640 x 480
	CASIA-IrisV4 [2] (Twins subset)	16212	640 x 480
	CASIA-IrisV4 [2] (Lamp subset)	3183	640 x 480
IKEMB-100	CASIA-IrisV4 [2] (Thousand subset)	20000	640 x 480
Cogent dual (CIS 202)	IIITD CLI [3]	3508	640 x 480
VistaFA2E single	IIITD CLI [3]	3075	640 x 480
IrisGuard AD100	Notre Dame Iris Cosmetic Contact Lenses 2013 [4]	900	640 x 480
LG4000	Notre Dame Iris Cosmetic Contact Lenses 2013 [4]	2800	640 x 480
LG4100	Notre Dame Iris Cosmetic Contact Lenses 2013 [4]	1400	640 x 480

Table 1.2. The dataset of iris camera models used in the experiments.

The ATVS-FIr DB[1] is an iris dataset from the ATVS Biometric Recognition Group. It was first made for liveness detection since it contains both real and fake examples form LG Iris Access EOU3000.

The CASIA-IRISV2 and CASIA-IRISV4[2] are provided by the biometrics research at Center for Biometrics and Security Research (CBSR), National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences (CASIA). The CASIA-IRISV2 includes two different devices the *Irispass-h*, developed by OKI and a self-developed device, named *CASIA-IrisCamV2*, while the CASIA-IRISV4 is collected by using a *Irispass-h* sensor and *IKEMB-100* dual-eye iris camera produced by IrisKing.

IIIT-D CLI database[3] [45, 46] is provided by Image Analysis and Biometrics Lab of the IIIT, Delhi. Iris images were captured by using two iris sensors: the *Cogent CIS 202* dual iris sensor and the *VistaFA2E* single iris sensor.

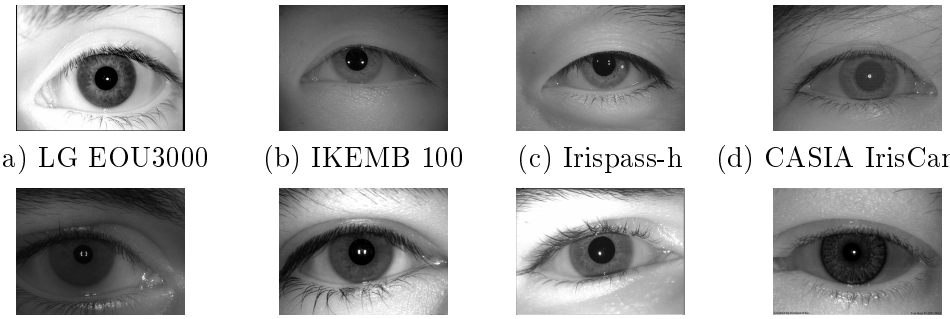
The Notre Dame Iris Cosmetic Contact Lenses dataset[4] [47], is provided by the Computer Vision Research Laboratory (CVRL) of the university of Notre Dame. This database contains iris images acquired by using an *LG4000*, an *LG4100* and an *IrisGuard AD100* iris sensor.

1.4.2 Experiments on natural images

In order to take into account a wide range of scenarios that might occur in real-world forensic applications, our camera identification tests are performed under various hypotheses:

- **Perfect knowledge:** there is a finite set of camera models (for example the 26 models of the Dresden Image Database) and we have

[1] http://atvs.ii.uam.es/fir_db.html
[2] CASIA Iris Image Database, http://biometrics.idealtest.org/
[3] http://www.iab-rubric.org/resources.html
[4] https://sites.google.com/a/nd.edu/public-cvrl/data-sets



(a) LG EOU3000 (b) IKEMB 100 (c) Irispass-h (d) CASIA IrisCamV2

(e) Vista Scanner (f) Cogent Scanner (g) LG4000 (h) AD100

Figure 1.2. Samples of iris images coming from different sensors of our dataset.

full knowledge about each of them. In practice, for each model we have a number of training images large enough to carry out reliable estimates of all the features of interest. Therefore, we can use a multiclass-SVM and will be able to compute a full confusion matrix.

- **Limited knowledge:** we have full knowledge on the target model, but know nothing about the number and features of other models. Practically, we have a large number of training images, classified as either belonging to the target model or another (unknown) model. Here, we can use only a one-class SVM, and will evaluate performance in terms of precision and recall.

- **Zero knowledge:** in this case we have no prior information on the number and features of camera models involved. There are only a large number of images, each with the associated feature. In this case, one can only hope to retrieve other images taken from the same model, which may help for subsequent investigations. Therefore, we will find the K nearest neighbors (NNs) to the target, and report on the precision@K, that is, the fraction of images that come from the

same model as the target among the first K retrieved NNs.

In the definition of the selected co-occurrence feature, we have left open the choice of the high-pass filter used to compute image residuals. Indeed, preliminary experiments show the performance to depend weakly on the specific filter. In any case, we will provide results for those that performed best in these tests, the s2_minmax_32 and s2_spam12hv, always with all the color components. In addition, we will provide results also for the s2_spam14hv filter, to allow comparison with the grayscale version used in our previous work [12]. Instead, we will not consider the use of denoising filters to compute residuals, since they increase significantly the computational complexity while not improving performance [12]. Finally, we implemented also the features proposed in Celiktutan-2008 [31], Gloe-2012 [34] and Xu-2012 [35]. Whenever possible, they are used as references for performance comparison. The *data-driven* approaches are compared in a dedicated section of *perfect knowledge* scenario while are not suitable for others where a full labeled dataset is not available.

1.4.2.1 Perfect knowledge case

Here, we assume to have a finite number of camera models, with an adequate number of training images for each model. Therefore, we train a multiclass-SVM with linear kernel, using 100 images for each camera model as training set, while all other images are used to test performance. When multiple devices are available for a single model, which holds in the majority of cases, all training images come from the same device, and no test images from the same device are used. Otherwise, training and test images come necessarily from the same device. We run this procedure 20 times, choosing each time at random the devices used for training. Eventually, results are averaged on all runs.

Results are reported in Table 1.3. The upper part of the table is for

Feature	length	accuracy	acc. w/crop
Celiktutan-2008	592	89.64	53.08
Gloe-2012	82	92.51	65.85
Xu-2012	354	98.15	94.59
s3_spam14hv (grayscale)	338	96.92	93.38
s3_spam14hv (color)	1878	97.21	93.85
s2_spam12hv (color)	1878	98.52	96.26
s2_minmax32 (color)	1875	**98.72**	95.70
s2_minmax32 (color)	1000	98.56	**96.27**
s2_minmax32 (color)	500	98.47	96.14

Table 1.3. Performance of various descriptors on the Dresden Image Database for whole images and for 512×512 crops.

reference methods, while the lower part concerns various versions of the features based on co-occurrences. The second column reports the feature length, while accuracy results are reported in column 3. All co-occurrence features perform very well, with the s2_minmax32 on all color bands, reaching almost 99% accuracy. Differences, however, are quite limited, speaking in favor of the approach independently of implementation details. Even the simpler grayscale feature (the best in [12]) grants a 97% accuracy. The best feature keeps working very well even after applying feature selection, with an algorithm based on Fisher score [48], and reducing the number of components to 1000 and 500 (last two rows of Table 1.3). It must be said that also some reference methods provide very good results, in particular Xu-2012.

In Table 1.3 we also report results computed on small crops taken from the images, i.e., 512×512 pixel sections which account for less than 5% of the whole image, on average. The co-occurrence features keep being highly reliable, with a performance loss of 2-3 percent points, while for all reference features the performance impairs more significantly, and dramatically so for Celiktutan-2008 and Gloe-2012.

Table 1.4 provides the full confusion matrix for the s2_minmax_32

feature. In the vast majority of cases the performance is perfect or near-perfect. Let us focus on the exceptions. There is a clear problem with some Sony models (DSC-H50 and DSC-W170): presumably, coming from the same manufacturer, they use the very same in-camera processing suite. The same considerations hold for the Canon Ixus55 and Ixus70, where, however, the problem is much more serious. Another critical case is the Nikon D200, which has been found relatively hard to identify also in other investigations, including [34], which raises interest on the in-camera processes it adopts.

In Table 1.5 we perform identification using JPEG compressed images. Indeed, uncompressed images are more the exception than the rule in the real world. Very often, images are JPEG compressed before being used, e.g., posted on a website or circulated on a social network. Therefore, it makes sense to repeat the identification experiment including compression, taking advantage of the fact that the compression quality factor (QF) can be easily estimated from the image itself. We therefore JPEG compress the whole dataset at the same QF, choosing QF=90, 75 and 60 to consider compression at various qualities, and repeat the very same identification experiment described before. As with the cropped images, we observe a very graceful degradation of performance as the QF decreases, with an accuracy close to 90% even at QF=60, for the best co-occurrence feature. Again, reference features present a sharper impairment of performance, and some of them are clearly unreliable with compressed images.

It is worth emphasizing again that in the above experiment the classifier is trained on images that are all JPEG compressed with the same QF as the target. Therefore, no information is provided on the robustness of features, but only on their versatility. On the other hand, robustness is certainly a desirable property, worth investigating. To this end we carried out a further experiment in which the target image is either compressed, at various quality factors QF, or resized, at various scales s, while the classifier is

	Identified as																												
Model	504	733s	830i	S505	S530s	I55	I70	A640	Z150	J50	M1063	S710	D200	D70	μ	FZ50	OA40	OW60	DCZ5	GX100	7325	L74w	NV15	H50	T77	W170
DC-504	98.55	-	-	-	1.45	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
DC-733s	-	99.45	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0.55	-	-	-	-	-	-
DC-830i	-	0.38	98.86	-	0.38	-	-	-	-	-	0.38	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
S505-X	-	-	1.39	98.61	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
S530s	-	1.10	-	-	97.43	-	-	-	-	1.47	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
I55	-	-	-	-	-	93.55	2.42	4.03	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
I70	-	-	-	-	-	1.61	98.39	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PSA640	-	-	-	-	-	1.14	-	98.86	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
EX-Z150	-	-	-	-	-	-	-	-	98.78	-	-	-	-	-	-	-	-	1.22	-	-	-	-	-	-	-	-
FPJ50	-	-	-	0.24	-	-	-	-	-	99.76	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
M1063	0.16	0.27	-	0.33	0.05	-	-	-	-	0.05	98.85	-	-	-	-	0.05	-	-	0.05	0.16	-	-	-	-	-	-
CPS710	-	-	-	-	-	-	-	-	-	-	-	99.60	-	-	-	-	0.40	-	-	-	-	-	-	-	-	-
D200	1.10	-	-	-	-	-	-	-	-	-	-	-	95.43	2.19	-	-	1.28	-	-	-	-	-	-	-	-	-
D70(s)	-	-	-	-	-	-	-	-	-	-	-	-	-	100.00	-	-	-	-	-	-	-	-	-	-	-	-
μ1050SW	-	-	-	0.49	-	-	-	-	-	-	-	-	-	-	97.81	0.49	-	1.22	-	-	-	-	-	-	-	-
DMC-FZ50	-	-	-	-	-	-	-	-	-	-	-	-	-	-	99.85	-	-	-	-	-	-	-	0.15	-	-	-
OA40	0.41	0.21	-	-	-	-	-	-	-	-	-	-	0.62	-	-	-	98.76	-	-	-	-	-	-	-	-	-
OW60	-	-	-	-	-	-	-	-	1.09	-	-	-	-	-	-	-	-	98.91	-	-	-	-	-	-	-	-
DCZ5.9	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	100.00	-	-	-	-	-	-	-
GX100	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0.30	-	-	99.70	-	-	-	-	-	-
RCP-7325XS	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	100.00	-	-	-	-	-	-
L74w	-	-	-	-	-	-	-	-	0.22	-	-	-	-	-	-	-	-	-	-	-	99.78	-	-	-	-	-
NV15	0.93	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	99.07	-	-	-	-
DSC-H50	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0.35	-	-	89.44	-	10.21	
DSC-T77	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	100.00	-	
DSC-W170	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0.49	-	-	11.71	-	87.80	

Table 1.4. Confusion matrix for the *s2 minmax32* feature.

Feature	length	accuracy			
		uncomp.	QF = 90	QF = 75	QF = 60
Celiktutan-2008	592	89.64	83.15	71.23	63.87
Gloe-2012	82	92.51	79.63	74.31	68.48
Xu-2012	354	98.15	94.85	88.91	82.90
s3_spam14hv (grayscale)	338	96.92	94.63	91.18	85.98
s3_spam14hv (color)	1868	97.21	95.84	93.06	88.22
s2_spam12hv (color)	1878	98.52	96.86	93.20	87.92
s2_minmax32 (color)	1875	**98.72**	**97.34**	**94.38**	**89.25**
s2_minmax32 (color)	1000	98.56	97.33	93.56	87.69
s2_minmax32 (color)	500	98.47	96.89	91.42	83.64

Table 1.5. Performance of various descriptors on the Dresden Image Database for uncompressed and JPEG compressed images at various QFs.

trained on the full quality images. The results, reported in Table 1.6, leave little room for interpretation. As soon as a mild compression (QF=90) or a moderate resizing (scale=90%) are applied the performance drops dramatically, becoming close to random guessing when more intense processing is applied. Such a behavior can be easily explained for co-occurrence based features, because they analyze micro-patterns in the image residual, which are certainly altered by compression and resampling. Even so, these feature show a higher (although always low) robustness w.r.t. most references.

1.4.2.2 Limited knowledge case

In the perfect knowledge case considered before a small universe of models is postulated. However, in the real world, a very large and steadily growing number of models is available. Accounting for all of them might be difficult, and certainly involves the managing of a very large image dataset with the corresponding storage and computational complexity problems. If also various forms of common processing are considered, the problem becomes soon unmanageable. Moreover, adding new models, requires intense work to update the classifier. Besides these practical problems, the

Feature	length	accuracy				
		original	QF=90	QF=60	s=90	s=50
Celiktutan-2008	592	89.64	25.88	11.27	35.43	6.54
Gloe-2012	82	92.51	27.85	18.80	17.15	6.72
Xu-2012	354	98.15	35.86	12.40	39.39	9.31
s3_spam14hv (grayscale)	338	96.92	24.42	10.48	39.85	8.21
s3_spam14hv (color)	1868	97.21	33.75	12.68	53.09	10.58
s2_spam12hv (color)	1878	98.52	34.96	16.26	43.07	12.36
s2_minmax32 (color)	1875	98.72	40.48	14.06	40.76	12.23

Table 1.6. Robustness of various descriptors on the Dresden Image Database w.r.t. compression and resizing.

multiclass approach, very appealing with a small number of classes, tends to become less and less reliable as the number of classes grows. In addition, forcing the target image to be associated with one of the available models gives rise to errors when the image comes instead from a different source. This is potentially dangerous, and plain unacceptable in many forensic applications.

Therefore, it makes full sense to consider an alternative scenario in which the prior knowledge is limited to just one camera model, and we test the H1 hypothesis that the target image fits such a model against the null hypothesis, H0, that it does not. Indeed, this approach was also explored in [34] in a similar setting. To gain insight into the importance of this case, consider an investigator looking in a large dataset for all the photos taken by the camera of a person of interests. By carrying out one-class model verification, most of (possibly all) the photos taken by the camera model of interest can be readily singled out, restricting and eventually speeding-up the search.

In our experiment, for each camera model of the Dresden Image Database we trained a one-class SVM using 100 training images coming from cameras of that model. All remaining images, both from the same model and

	precision	recall	accuracy
original image	84.50	68.76	98.46
512×512 crop	76.10	67.60	97.99
JPEG QF=75	62.45	53.32	96.03

Table 1.7. Performance in the limited knowledge case for the *s2 minmax32* (color) feature.

from other models, are then used as test set. Relevant parameters are kept fixed for all models. Results are reported in Table 1.7, only for the best feature, in terms of accuracy, precision and recall, as usual for these kinds of decision problems. They are computed for each camera model, and eventually averaged on all of them.

Working on uncropped and uncompressed images, precision and recall are pretty good but certainly not perfect, with over 30% of the images of interest that go undetected, and about a 15% of false positives. To gain insight on this latter result, however, consider that our test set is highly unbalanced, with most images taken by cameras of other models. In these conditions, having only 15% of false positives is actually quite remarkable. This is also reflected by the very high overall accuracy. Instead, a 30% of misses might be a problem in some cases. Of course, depending on the application of interest, one can modify the classification threshold to improve recall, for example, at the expense of precision. Turning to images that are either cropped (to 512×512 pixels) or JPEG compressed (with QF 75) we observe (last two lines of the table) a slight decrease in performance in the first case, which becomes more significant in the second one.

Table 1.8 reports results obtained in the same conditions as before but using feature selection to reduce the components from 1875 to 500. The performance does not change much when uncompressed images are considered, either complete or cropped, with minor shifts between precision

	precision	recall	accuracy
original image	92.38	64.08	98.50
512×512 crop	73.23	73.32	98.06
JPEG QF=75	50.28	42.28	95.25

Table 1.8. Performance in the limited knowledge case for the *s2 minmax32* (color) feature with feature selection (500 instead of 1875 components).

and recall. On the contrary, the effect is significant on JPEG compressed images, with recall going below 50% and precision just above.

1.4.2.3 Zero knowledge case

We conclude this analysis by considering the situation in which we have no clue on the camera used to take the photo of interest, but still want to retrieve some useful information from the available large dataset of images. In practice, the only action possible, based on the extracted feature, is to retrieve other images that share a very similar feature, thus reducing the search space to images presumably originated by cameras of the same model, see figure 1. Clearly, this is not a model identification problem anymore, but this experiment sheds further light on the expressivity of the selected feature.

In detail, we use the feature extracted from the available image as a query, and look for the K nearest neighbors features in the dataset, using a kd-tree based search algorithm. The performance is measured by the precision@K, the percentage of hits over the first K nearest neighbors, considering as hits all images that belong to the same model as the query, and averaging results over all images of the dataset used in turn as queries.

For the by now usual s2_minmax32 (color) feature we observe 81.00% hits over the top 10 neighbors (K=10), which decreases to 70.68% for K=20, 55.77% for K=50, and 44.16% for K=100 (1.9). Experiment on

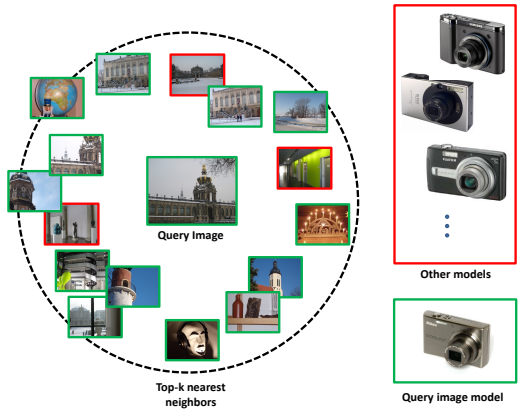


Figure 1.3. In the absence of prior knowledge, the co-occurrence extracted from the target image can be used to query the dataset and retrieve photos taken by the same camera model.

Feature	length	accuracy MAP@10	MAP@20	MAP@50	MAP@100
Celiktutan-2008	592	45.50	35.88	26.42	21.17
Gloe-2012	82	36.78	29.39	22.43	18.32
Xu-2012	354	83.03	72.92	59.11	49.29
LBP256_color_f33	768	86.08	79.39	69.47	60.66
s3_spam14hv	338	84.09	75.37	62.59	51.93
s3_spam14hv(color)	1868	82.95	73.83	60.60	49.96
s2_spam12hv(color)	1878	85.17	76.32	63.01	51.85
s2_minmax32(color)	1875	81.00	70.68	55.77	44.16

Table 1.9. Mean average precision of various descriptors on the Dresden Dataset for original images, in the zero knowledge scenario using a kd-tree search algorithm.

compressed images are also reported in Fig.1.10. However limited, this experiment shows clearly that the co-occurrence feature allows one to retrieve a large number of images taken from the same model as the query. Of course, if the retrieved images have camera information attached with them, this would again allow for an indirect identification.

Feature	length	accuracy			
		MAP@10	MAP@20	MAP@50	MAP@100
Celiktutan-2008	592	30.12	24.71	18.32	15.37
Gloe-2012	82	28.26	22.15	17.26	14.54
Xu-2012	354	55.34	42.21	29.91	23.59
LBP256_color_f33	768	48.74	37.72	27.33	21.75
s3_spam14hv	338	52.00	40.11	28.82	22.78
s3_spam14hv(color)	1868	52.96	40.92	29.48	23.47
s2_spam12hv(color)	1878	54.58	42.28	30.42	24.00
s2_minmax32(color)	1875	47.23	35.89	25.67	20.38

Table 1.10. Mean average precision of various descriptors on the Dresden Dataset for JPEG compressed images at QF=75, in the zero knowledge scenario using a kd-tree search algorithm.

1.4.2.4 Comparison with CNN-based feature

In this section we show the experiment performed on Dresden dataset in *perfect knowledge* scenario with the use of CNN-based approach proposed. We compare the results with the CNN in [15], that use a CNN followed by an SVM pipeline for the camera model identification. The comparison is performed respect to the two state-of-the-art methods that performs better: our proposed best feature *s2_minmax32* working on color images, and LBP based feature proposed by Xu [35]. Since, to train a CNN we need both a train and a test set, only a *perfect knowledge* scenario is considered. Is worth to note that our proposed network works on grayscale images while all the comparison method exploit the full color images.

The comparison highlights that CNN-based methods outperform both [12] than [35] approaches on small of size 64×64,128×128 and 256×256, patches. The best CNN method in classifying single image patches give an overall accuracy over 93% even using only a 64×64 patch where the performance of *blind* features decrease significantly. Our proposed method do not perform better than one in [15] but still gives comparable result

Feature	64×64	128×128	256×256	Full
Xu-2012	54.53	76.64	89.01	98.15
Bondi-2017	92.82	–	–	–
Proposed co-occurence (color)	67.21	84.17	91.21	98.72
Proposed CNN	91.63	93.15	94.15	–

Table 1.11. Performance of CNN-based descriptors on the Dresden Image Database for different size of crops.

even using a simpler net.

1.4.3 Experiments on iris sensor images

In this section we show that the model identification approaches work even on images taken in different contest, such as images coming from iris sensor. The importance of having a good camera model identification method for iris sensor is first assessed in [49], where the interoperability problem of iris sensor in the same system is explicitly decoupled in its two basic components, namely, *i)* identifying sensor models, and *ii)* mapping images, or extracted features, from one sensor to the other. In [49] the model identification relies on some global features computed in the wavelet domain, including selected means, variances and entropies. To take into account technological constraints, the relatively small architecture proposed is used, thus limiting both computational complexity and memory requirements. Moreover, transfer learning is used to speed-up training and reduce the training set size. Here we report only the experimental results and the comparison made in the original paper [17].

Experiments performed on a large number of iris images coming from different databases prove that the proposed solution improve model identification performance with respect to the reference methods. Moreover, even if not reported is worth to note that keeping the structure of the iris recognition system proposed in [49], the more reliable identification step

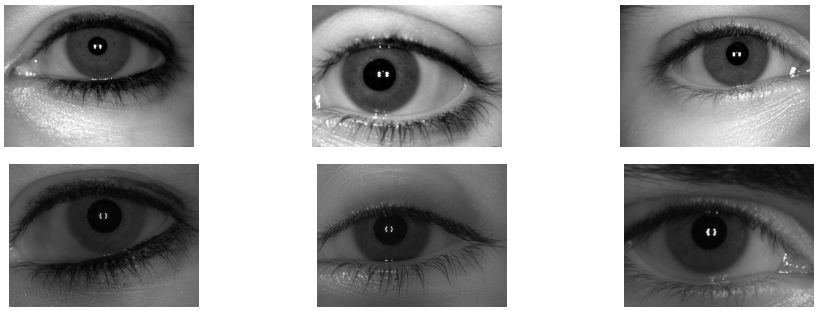


Figure 1.4. Iris images captured by different sensors and at different distances. The first and the second row are iris images captured by different sensors. Images in the first row are acquired by the Cogent dual iris sensor, images in the second rows are acquired by the VistaFA2E sensor. In the first two rows, images of the same columns belong to the same subject.

impacts on the overall performance of the biometric recognition system, so improving sensor interoperability.

1.4.3.1 Iris model identification results

The iris model identification approaches are evaluated by using all the nine iris sensor models described in the previous Section. In particular, 900 samples were randomly selected from the database of each model, in order to have a dataset with the same number of images per model, made up by a total of 8100 iris images. 60% of this dataset was used as training set and the remaining 40% for testing, taking care of selecting iris classes (subjects) in the test set different from those present in the training set. In order to better train the CNN architecture proposed and the reference [15] a data augmentation was performed, by extracting 9 patches from the center of each image in the original training set, as illustrated in Fig. 1.5. Three different sizes of the patches have been considered, namely 256×256 pixels, 128×128 pixels and 64×64 pixels. Moreover, two different training settings have been considered for our proposal. In the first one (*training form*

Feature	64×64	128×128	256×256	Full
Arora-2012	–	–	–	91.80
Xu-2012	91.73	93.81	97.21	99.04
Bondi-2017	94.14	–	–	–
Proposed co-occurence (grayscale)	92.49	94.54	96.43	98.75
Proposed CNN (*from scratch*)	95.08	98.13	99.29	–
Proposed CNN (*finetuning*)	97.18	98.57	99.35	–

Table 1.12. Results of different model identification approaches on the test set by varying the size of input images.

scratch) the CNN is trained anew on the training data; in the second one (*Dresden finetuning*) a net pre-trained on images coming from the Dresden database [44] was used. Results of the proposed CNN-based architectures are reported in table 1.12 together with those obtainable by the approach presented in [49], and by using a SVM classifier with a linear kernel on features extracted according to the approaches proposed in [35] and [16]. It is worth noting that while the features used by the approaches under comparison have been extracted from the whole iris images, in case of our CNN-base architectures, we considered crops of the images relative to 256×256 pixels, 128×128 pixels and 64×64 pixels. In order to have a fairer comparison, we also report the results obtained by using the features proposed in [35] and [16] extracted on crops of the same size. As regards the approach proposed in [49], it must be observed that it cannot perform well on crops coming from the original images. The CNN proposed by [15] is shown only for patch of size 64×64 as in the original paper.

As it is evident from table 1.12, the CNN-based approach that use 256×256 pixels crops perform better than the comparing approaches. The improvement with respect to the approach presented in [49] is also statistically significant. When small crops are used for training, the *Dresden finetuning* version of the CNN is able to outperform the *training-from-the-scratch* one. In this case the improvement with respect to the other

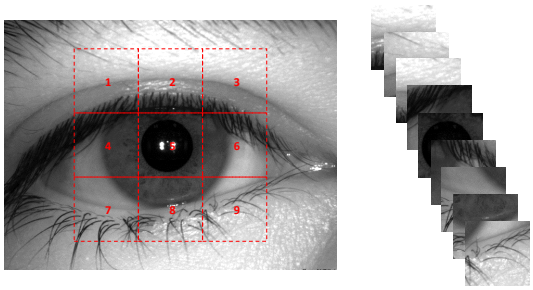


Figure 1.5. Example of cropping iris images for data augumentation. In this case we obtain nine 128x128 non-overlapping patches. Note that in case of bigger crops the patches can overlap each other.

feature-based approaches that use images of the same size for training also becomes significant. Unlike the results obtained for natural images, here the CNN proposed in [15], do not perform better. One of the reason could be the size of dataset, that is smaller than before and is not suitable for training such a bigger net.

Table 1.13 shows the confusion matrix relative to the best result obtained by the proposed CNN-based architectures. Images coming from 6 models out of 9 are perfectly recognized, while the most part of errors are due to misrecognitions between two very similar models coming from the same vendor (LG4000 and LG 4100), so confirming the validity of the proposed approach.

Model		EOU	IKEMB	OKI	ICV2	COG	FA2E	AD100	4000	4100
LG Iris Access EOU3000	EOU	100	-	-	-	-	-	-	-	-
IKEMB-100	IKEMB	-	100	-	-	-	-	-	-	-
OKI IRISPASS-h	OKI	-	-	100	-	-	-	-	-	-
CASIA-IrisCamV2	ICV2	-	-	-	99.3	-	-	-	0.4	0.3
Cogent dual	COG	-	-	-	-	100	-	-	-	-
VistaFA2E single	FA2E	-	-	-	-	-	100	-	-	-
IrisGuard AD100	AD100	-	-	-	-	-	-	100	-	-
LG4000	4000	-	-	-	-	-	-	-	95.9	4.1
LG4100	4100	-	-	-	-	-	-	-	0.9	99.1

Table 1.13. The confusion matrix obtained by the proposed approach.

Chapter 2

Blind source clustering

In blind source identification scenario, there is no prior information on the number and kind of possible devices involved. The analyst is given only a number of images, with no useful metadata attached with them, which might or might not be related with one another. In this scenario, classical identification is not viable anymore. Nonetheless, it may be important to understand which images come from the same camera, and which are not. Lacking any other clues, useful information may be extracted only through image clustering. In fact, if the images have been taken by just a few cameras, they can be clustered according to their common unknown source. Besides linking photos with one another, clustering allows one to estimate some important features of the unknown cameras, like their PRNU, which can be used to identify new images coming from different repositories.

A few examples may provide some insight into the importance of this task. Once gained access to a hard disk with child abuse images, investigators may use blind clustering to estimate the number of devices used to take the photos and their fingerprints. These may allow one to trace back, by further analyses, all authors of the photos. On a larger scale, law en-

forcement agencies may download a large number of such images from the net, and use blind clustering followed by fingerprint estimation to build a database of PRNU patterns to use in future investigations or even in court. With reference to this last task, a growing number of papers pursue user identification or image-to-identity linking over social networks [50, 51, 52]. *Correlation clustering* (CC) is a recently proposed method for data partitioning. Given a suitable measure of data similarity (correlation), the optimal partition is obtained by solving a constrained energy minimization problem. Integer linear programming tools ensure fast computation. Like all clustering algorithms, CC depends critically on the sensible setting of some parameters. In order to overcome this need, we use *consensus clustering* [53, 54] to extract a unique solution which aggregates a number of base CC clusterings. As a result of these steps, we obtain a first conservative partition, characterized by a very low probability of finding unrelated residuals in the same cluster. Then we proceed with an *ad hoc refinement algorithm* which iteratively merges clusters. As the algorithm proceeds, larger and larger clusters emerge, leading in turn to better estimates of the corresponding PRNUs, and allowing the inclusion of further small clusters and outliers, until a suitable stopping condition is met. We already proposed the use of correlation clustering and cluster refinement in our recent work [55]. However, in that algorithm some key parameters had to be set by the user based on preliminary analyses on a suitable training set. This is a nuisance for the user and a source of impairment related to estimation errors. In this thesis, I show my work presented in [56] that modify substantially the algorithm of [55] by *i)* introducing the consensus clustering step, and *ii)* adopting a new parameter-free probabilistic merging criterion. By so doing, we obtain a fully unsupervised clustering tool, where no parameter must be set by the user or estimated on external training set.

In the rest of the chapter we will discuss state of the art work on

Blind Source Clustering (Section II), provide the necessary background and notation for the proposed method (Section III), describe the proposed algorithm (Section IV), and finally we describe experiments and results (Section V)

2.1 Related work

PRNU-based image clustering has necessarily a short history. To the best of our knowledge, Bloy has been the first researcher to deal with this problem, proposing in [8] a modified version of the pairwise nearest neighbor (PNN) algorithm [57]. In PNN, the source data are regarded as single-point clusters. The distances between all pairs of clusters are computed, then the closest pair is merged into a new cluster, represented by its centroid, and the process goes on recursively, until a suitable stopping condition is met. Clearly, as clustering proceeds, better and better PRNU estimates become available. To speed up the process, Bloy introduced some modifications to the original PNN, like picking at random couples of clusters candidate for merging, or looking for all neighbors of a cluster before proceeding with another one. Besides the need for the user to provide a sensible threshold to decide on merging, a major drawback of this method is its computational burden. Several variants of this procedure can be found in the literature, aimed at reducing its computational cost or improving accuracy. Sometimes, a pre-processing step is introduced in order to enhance the fingerprint [58, 59]. In [60] a faster solution based on hierarchical clustering is proposed, together with a criterion based on a silhouette coefficient [61], which combines measures of cohesion (inside clusters) and separation (among clusters). The same solution, with minor modifications, is followed in [62] for the purpose of smartphone clustering. In [63], the original camera fingerprints are replaced by their compressed version. The time saved on PRNU-related computation, is used to re-

store a non-random search phase, where the two clusters with the highest correlation are selected as candidates at each iteration. Finally, in [64] a refinement step based on Hu's moment vector is applied in order to improve performance. A different approach is followed in [65], where PRNU-based clustering is regarded as a graph partitioning problem. Each image is considered as a node in a weighted undirected graph and the nodes are partitioned into disjunct sets by means of spectral analysis [66]. Thanks to the use of efficient graph processing tools, this algorithm is much faster than those based on PNN. On the down side, the performance depends strongly on the random initialization, and the number of clusters must be set in advance by the user. These problems are partially solved in [67] by adopting the Normalized Cuts graph partitioning algorithm [68], which guarantees a more stable performance and does not need the number of clusters as input parameter. However, the stopping criterion is based on the comparison of an aggregation coefficient with a suitable threshold, a critical input parameter itself. In [67] the optimal threshold value is estimated by preliminary experiments on a training set, but no guarantee can be given on the alignment of training and test sets.

Regardless of performance, a common undesirable trait of all these algorithms is the need for the user to specify or estimate some critical parameters in advance, which may prevent their use in practical applications.

2.2 Background

In this Section we provide the background and notation necessary to ensure the self-consistency of the paper and guarantee the full understanding of the proposed method. After recalling some concepts and results on PRNU-based identification, we will briefly describe correlation clustering, and finally consensus clustering, with special reference to the recently proposed [69] WEAC (weighted evidence accumulation clustering) algorithm.

2.2.1 PRNU

The photo-response non uniformity (PRNU) is an intrinsic and stable characteristic of each individual camera, caused by tiny imperfections in the manufacturing process of the sensor. Following the simplified model of [4, 70], the image observed at the camera output, I, can be written as

$$I = (1 + K)I^{(0)} + \Theta \tag{2.1}$$

(products between images, unless otherwise stated, are pixel-wise) where $I^{(0)}$ is the ideal noise-free image, K the camera PRNU, and Θ an additive noise term which accounts for all types of disturbances. Typically, $K_s \ll 1$ for all non-faulty pixels, s, of the image, hence the disturbance produced by the PRNU is usually unnoticed. Still, since each image acquired by a given camera contains traces of its PRNU pattern, this can be used for various forensic applications, from source identification [2, 71, 72] to forgery detection and localization [4, 73, 74, 75]. The PRNU can be estimated by extracting the noise-free image by means of a denoising filter f

$$\widehat{I}^{(0)} = f(I) \tag{2.2}$$

and removing it from the acquired image, to obtain the so-called noise residual

$$\begin{aligned} R &= I - \widehat{I}^{(0)} \\ &= IK + (I^{(0)} - I)K + (I^{(0)} - \widehat{I}^{(0)}) + \Theta \\ &= IK + \Theta' \end{aligned} \tag{2.3}$$

with the new noise term, Θ', accounting also for denoising errors, $I^{(0)} - \widehat{I}^{(0)}$. Given M images acquired by the camera of interest, one can perform a

maximum-likelihood (ML) estimate of the PRNU as [4]

$$\widehat{K} = \sum_{i=1}^{M} \left[\frac{I_i}{\sum_{i=1}^{M} I_i^2} \right] R_i \tag{2.4}$$

followed by a further step to remove non-unique artifacts originated by other internal camera processes. Note that, to obtain a reliable estimate, a large number of input images is required, because the PRNU component, even in the most favourable conditions, is much weaker than the noise component. Given an estimate of a camera PRNU, several tests can be used to decide whether an image, I, comes from the same camera. The most popular one is based on the correlation index, $\mathrm{corr}(R, I\widehat{K})$, between the image residual, R, and the scaled PRNU, where

$$\mathrm{corr}(x, y) = \frac{\langle (x - \overline{x}), (y - \overline{y}) \rangle}{\|x - \overline{x}\| \cdot \|y - \overline{y}\|} \tag{2.5}$$

$\langle x, y \rangle$ denotes the inner product between x and y, $\overline{x}$ indicates the mean of x, and $\|x\|$ its Euclidean norm. Another common statistic is the peak-to-correlation energy ratio (PCE), computed in the Fourier domain, which is more robust to image cropping.

In this paper, rather than the ML estimate, we will use the sample mean

$$\widetilde{K} = \frac{1}{M} \sum_{m=1}^{M} R_i \tag{2.6}$$

and the related decision statistic $\mathrm{corr}(R, \widetilde{K})$. Moreover non-unique artifacts are removed from the beginning from each residual. By doing so, both the PRNU estimates and the correlations can be computed without reference to the original images and without post-processing, which entails important algorithmic simplifications and, eventually, high computational efficiency. On the other hand, the sample-mean estimate of the PRNU

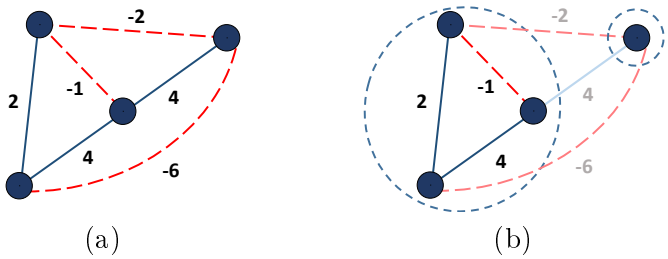


Figure 2.1. Correlation Clustering applied to a toy example. Left: initial graph; links with positive[negative] weight are shown in blue[red]. Right: an acceptable two-cluster partition; due to coherence constraints, a link with negative weight (-1) is kept in the final partition, and one with positive weight (4) is cut.

is largely adopted in the literature and shown [76] to be equally effective than the ML estimate.

2.2.2 Correlation clustering

Clustering can be cast as a graph partitioning problem. Let $G = (V, E)$ be an undirected graph, where V is the set of nodes, to which the data instances are associated, and E a set of edges, possibly incomplete, connecting couples of nodes. The goal of graph partitioning is to create several disjoint clusters of nodes according to some suitable criterion. Therefore, all edges linking nodes of different clusters must be cut, while the others may be kept. In correlation clustering [77], a weight w_{ij} is associated with each edge $e = (i, j)$, expressing the *correlation* between nodes i and j. Although the precise meaning of correlation depends on the specific problem, a positive[negative] correlation indicates, in general, that the linked nodes are desired to belong to the same[different] cluster. Based on this information, one may be tempted to cut all nodes with negative weights and keep the remaining ones. This simplistic approach, however, gives rise easily to incoherent solutions. In fact, two nodes can be linked through

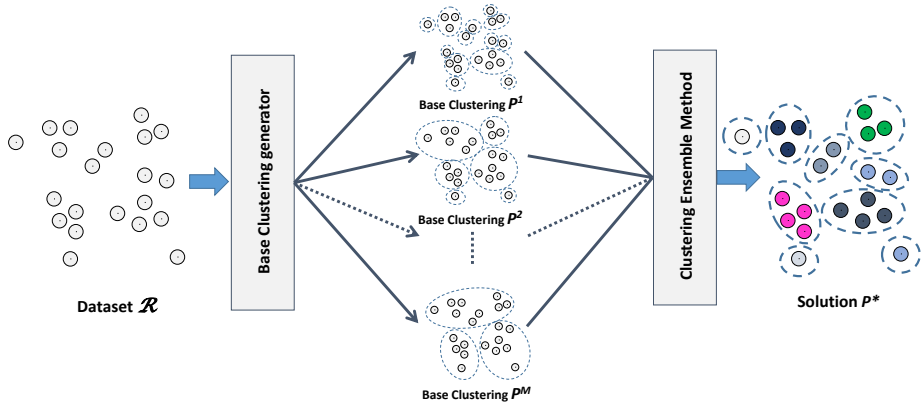


Figure 2.2. The ensemble clustering problem.

multiple paths of various lengths, and all such paths must be cut at some points to ensure the two nodes actually belong to different clusters. As an example, for the simple graph shown in Fig.2.1 on the left, the two-cluster solution shown on the right requires cutting a link with positive weight and keeping a link with negative weight. Therefore, correlation clustering formulates the graph partitioning problem as a constrained energy minimization. Let x_e be the binary indicator variable specifying whether edge e is cut ($x_e = 1$) or retained ($x_e = 0$), and $\mathbf{x} \in \{0,1\}^{|E|}$ a generic configuration of the edges. With each configuration, an energy $\mathscr{E}(\mathbf{x})$ is associated, defined as the sum of the weights of all cut edges

$$\mathscr{E}(\mathbf{x}) = \sum_e w_e x_e \tag{2.7}$$

In pursuing the minimization of this energy, however, the coherence constraints must be taken into account. If nodes i and j are to belong to different clusters ($x_{ij} = 1$), any other node k cannot be grouped simultaneously with both i and j ($x_{ik} + x_{jk} \geq 1$). These constraints may be

expressed synthetically [77] as

$$x_{ij} - x_{jk} - x_{ik} \leq 0 \quad \forall\, i, j, k \tag{2.8}$$

and only the configurations respecting these constraints, $\mathbf{x} \in X_c$, correspond to acceptable solutions. Eventually, the problem can be expressed as

$$\mathbf{x}^{CC} = \arg\min_{\mathbf{x} \in X_c} \mathscr{E}(\mathbf{x}) = \arg\min_{\mathbf{x} \in X_c} \sum_{e \in E} w_e x_e \tag{2.9}$$

Note that, since the coherence constraints are linear, the optimal graph partition can be found by resorting to Integer Linear Programming (ILP). The problem however is NP-hard, and hence, for large graphs, finding the exact solution may become exceedingly complex. Consequently, greedy techniques are typically adopted [78], which provide slightly sub-optimal solutions but in a much shorter time.

2.2.3 Consensus clustering

A major problem with data clustering is that results depend significantly on the selected algorithm and, even for a given algorithm, on some critical parameters, like the number of clusters or some decision thresholds. Therefore, by running multiple algorithms, or even just one algorithm with different parameters, one ends up with a number of alternative, and possibly very different, clusterings. However, this over-abundance of solutions, if properly exploited, represents a precious source of information. The goal of consensus clustering is to suitably combine all these clusterings to provide a unique and more satisfactory solution, as shown pictorially in Fig.2.2.

Let

$$\mathscr{R} = \{R_1, R_2, \ldots, R_n\} \tag{2.10}$$

be the dataset under analysis, with n data points. A partition P^i of the

dataset is a collection of n_i disjoint clusters of data

$$P^i = \{C_1^i, C_2^i, \ldots, C_{n_i}^i\} \tag{2.11}$$

with $C_j^i \bigcap C_k^i = \emptyset$, $\forall j \neq k$, and $\bigcup_{j=1}^{n_i} C_j^i = \mathscr{R}$. By running one or more clustering algorithms, with different choices of the parameters, we obtain M such partitions, or *base clusterings*,

$$\mathscr{P} = \{P^1, P^2, \ldots, P^M\} \tag{2.12}$$

which are jointly processed to extract eventually a single consensus clustering P^*. In the literature, several approaches have been proposed for consensus clustering with a large number of specific algorithms [54, 79]. A first approach is to formulate ensemble clustering as a graph partitioning problem, as originally done in Strehl [80] and then in [81], where the graph formulation simultaneously models both instances and clusters of the ensemble as vertices in a bipartite graph. Recently, a more robust algorithm has been developed in [69] based on sparse graph representation and probability trajectory analysis. In [82], instead, the Weighted Spectral Cluster Ensemble (WSCE) method is proposed where a new version of spectral clustering is considered together with a specific solution for combining the individual clustering results. A flexible and computationally scalable approach is proposed in [83], where a Bayesian framework is developed for simultaneous estimation of both the consensus clustering and the source-specific clusterings. A further set of methods rely on pair-wise similarity, and the Evidence Accumulation Clustering (EAC), proposed in [53], is among the most popular methods following this approach. It is based on a co-association matrix which counts how many times two objects occur in the same cluster in the ensemble of multiple base clusterings. In the following we will describe in more detail this method and its recently proposed generalization [69] adopted in the proposed algorithm. Let $P^i(R_j)$

indicate the cluster associated with R_j under the i-th partition. We define the co-occurrence similarity matrix S^i associated with partition $P^i \in \mathscr{P}$ as the matrix with elements

$$S^i_{j,k} = \begin{cases} 1 & \text{if } P^i(j) = P^i(k) \\ 0 & \text{otherwise} \end{cases} \tag{2.13}$$

That is, entry (j,k) of the matrix is 1 if R_j and R_k belong to the same cluster under P^i, and 0 otherwise. We then build the association matrix A by averaging the similarity matrices over all base clusterings

$$A = \frac{1}{M} \sum_{i=1}^{M} S^i \tag{2.14}$$

Therefore $A_{j,k}$ goes from 0, when data points R_j and R_k never belong to the same cluster, to 1, when they always do. Based on this "evidence", collected over the whole set of base clusterings, one can apply a single-linkage agglomerative clustering method [84] to obtain the final consensus clustering P^*. Note that this latter algorithm needs in input the final number of clusters n^*, which must be known *a priori* or estimated by other means. Experimental evidence [53] shows that EAC (like other consensus clustering algorithms) provides, typically, a significant performance improvements over *all* base clusterings. Occasional errors may be originated by outliers, base clusterings very different from the others, which impair the quality of the association matrix. To avoid such problems, we adopt a recently proposed [69] generalization of EAC, the Weighted Evidence Accumulation Clustering (WEAC) which computes the association matrix using generic weights

$$A = \sum_{i=1}^{M} w_i S^i \tag{2.15}$$

with $\sum_{i=1}^{M} w_i = 1$. The weights are chosen so as to prevent outliers from

affecting significantly the result. This is obtained by relying on the "wisdom of the crowd", a widespread concept in social and economic sciences [85] by which the most popular opinions are also the most relevant. Each base clustering is considered as an opinion and compared with all the other clusterings through a suitable similarity measure, $\mathrm{sim}(P^i, P^j)$, to compute a crowd agreement index

$$\mathrm{CAI}_i = \frac{1}{M-1} \sum_{j=1, j\neq i}^{M} \mathrm{sim}(P^i, P^j) \tag{2.16}$$

These indexes are then used to compute the final weights as

$$w_i = \frac{\mathrm{CAI}_i^{\beta}}{\sum_{j=1}^{M} \mathrm{CAI}_j^{\beta}} \tag{2.17}$$

where the exponent β is used to further emphasize differences. Following again [69] we set $\beta = 2$, and use the normalized mutual information as similarity measure.

A final problem is the choice of the final number of clusters n^*. Although several methods have been proposed in the literature [53, 86] we will use a new heuristic, explained in next Section, which better fits the needs of our specific problem.

2.3 Two-step source clustering

Before describing the details of the proposed method we want to motivate our design choices in light of the peculiar features of PRNU-based clustering, and of our goal to obtain a totally blind no-reference algorithm.

Our elementary data are the noise residuals, $R_1, \ldots, R_n$, and we cluster them based on their similarity, measured, for residuals R_i and R_j, by the

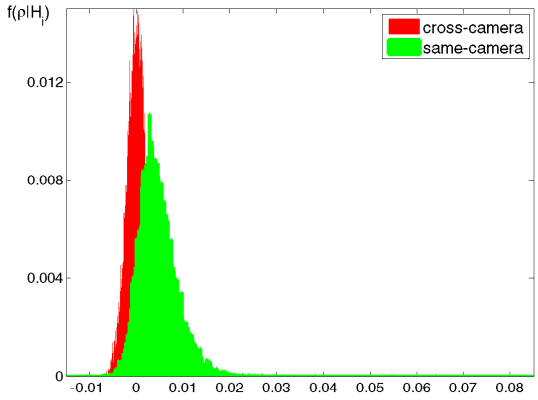


Figure 2.3. Distribution of cross-camera (red) and same-camera (green) correlations. The correlations are computed among individual noise residuals.

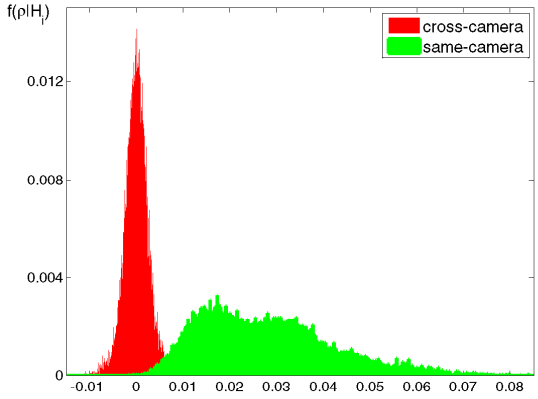


Figure 2.4. Distribution of cross-camera (red) and same-camera (green) correlations. The correlations are computed among individual noise residuals and camera PRNUs estimated on 50 residuals.

correlation index

$$\rho_{ij} = \mathrm{corr}(R_i, R_j) \tag{2.18}$$

Residuals from the same device (same-camera) exhibit, on average, a larger correlation than residuals from different devices (cross-camera). However, due to the strong noise component, the distributions of same-camera and cross-camera correlations overlap to a large extent, as shown in Fig.2.3, raising serious doubts on the chances of ever obtaining a reasonable result. In our problem however, as clustering proceeds (correctly), more and more same-camera residuals are aggregated, providing increasingly better PRNU estimates. These, in turn, can be used to reliably select new residuals to include in the cluster, giving rise to a virtuous circle. Fig.2.4 provides evidence to support this reasoning. In fact, same-camera and cross-camera correlations computed between individual residuals and 50-residual PRNU estimates exhibit well separated distributions.

This preliminary analysis makes clear that clustering and estimation should proceed hand in hand. In fact, this is exactly what happens with some well-known clustering methods, like the PNN with its iterative cluster merging. Yet, we follow a different path, carrying out a preliminary correlation clustering, with no estimate updating, and adopting the estimate/merge alternation only in the final refinement phase. The reason is that correlation clustering, through its constraints, takes into account all relationships at once. The decision on whether to merge two data points in the same cluster or keep them separate depends, heavily, also on the correlation of both points with third parties. A good example is provided again by Fig.2.1. Indeed, one might think of merging the rightmost node with a close one because of their positive (+4) link. This merging, however, must be ultimately rejected because of the negative links (-6 and -2) with other nodes of the same cluster. Overall, a much wider information base is taken into account than in pairwise processing, allowing one to avoid unwise decisions with long-term effects. Using WEAC on top of correlation clustering allows us to exploit the "wisdom of the crowd" and improve the performance of this initial step. In addition, it frees the user from the

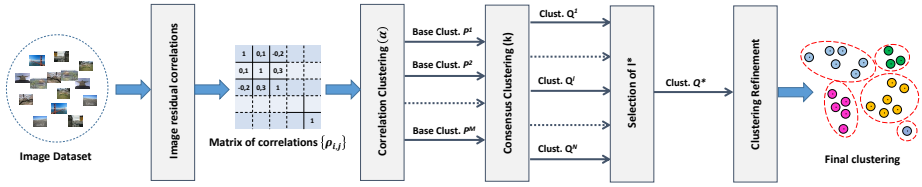


Figure 2.5. Block diagram of the proposed method. Block 1 computes correlations among image residuals: the matrix of correlations is the main data structure of the algorithm. Block 2 (correlation clustering) outputs a different base partition for each value of parameter α. Block 3 (consensus clustering) outputs a different consensus partition for each value of the number of clusters. Block 4 selects the optimum partition Q^* among these. Block 5 (cluster refinement) merges clusters until convergence.

need to set a critical parameter, α, described later on, moving a decisive step towards a fully unsupervised algorithm.

Starting from relatively small, but highly reliable, initial clusters, we run an *ad hoc* refinement phase based on the estimate/merge alternation mentioned before. To decide which clusters to merge at each step, suitable maximum-likelihood ratio statistics are computed, based again on residual correlations, and the same statistics provide the stopping condition of the algorithm.

These processing phases are now described in detail in the following subsections. summarized pictorially in Fig.2.5.

2.3.1 Pre-processing

As a preliminary step, we compute the noise residuals of all images under analysis

$$R = g(I - f(I)) \tag{2.19}$$

where $f(\cdot)$ indicates image denoising, and $g(\cdot)$ the processes used to remove the mean and non-unique artifacts. In our implementation, denoising is performed by means of the BM3D filter [87], which proved very effective

[13] for the purpose of PRNU estimation. Non-unique artifacts are removed as suggested in [4], that is, by zero-meaning all columns and rows, and by Wiener filtering in the Fourier domain.

Afterwards, we compute and store the inner products between all couples of residuals

$$c_{ij} = \langle R_i, R_j \rangle \tag{2.20}$$

and the related correlation indexes

$$\rho_{ij} = \frac{c_{ij}}{\sqrt{c_{ii}}\sqrt{c_{jj}}} \tag{2.21}$$

After this preliminary step, neither the original images nor the residuals themselves will be used anymore, as all items of interest can be computed based on the matrix of inner products. This will speed-up tremendously the subsequent steps, and in fact, the pre-processing accounts for the largest part of the overall computational complexity.

2.3.2 Correlation clustering of noise residuals

We run correlation clustering using a fully connected graph, where the residuals are associated with the vertices, and the weights account for the correlation between them. Following [55], we define the weights as

$$w_{ij} = \rho_{ij} - \alpha \tag{2.22}$$

The constant α plays a crucial role in the algorithm, as it defines which weights are positive and which are negative. Remember that a positive[negative] weight w_{ij} implies a tendency to merge[separate] residuals R_i and R_j. Hence, $\alpha \gg 0$ implies no clustering at all, while $\alpha \ll 0$ leads to a single cluster including all residuals. Contrary to intuition, setting $\alpha = 0$ is not a good choice. To gain insight into this problem, let us assume, as done in [4] and also in [55], that cross-camera correlations (hypothesis H_0)

have a zero-mean Gaussian distribution

$$\rho|H_0 \sim N(0, \sigma_0) \tag{2.23}$$

In this case, using $\alpha = 0$, half the weights between cross-camera residuals would be positive, leading to a large number of undesired mergings and, very likely, to less clusters than cameras. In [55], to obviate this problem, after estimating the standard deviation σ_0, we set $\alpha = 3\sigma_0$. This is a rather conservative choice, by which cross-camera links have almost always a negative weight, $\Pr(w>0\,|\,H_0) \simeq 10^{-3}$, and the same happens also for many same-camera links, due to the overlapping distributions, leading to many more clusters than cameras. However, in this first step of the process, over-clustering is largely acceptable, since it will be corrected by the subsequent refinement step, which operates only through cluster merging. On the contrary, under-clustering, with the emergence of clusters with cross-camera residuals, represents an unrecoverable error. In this work we do not fix α anymore. Instead, we perform CC with a large number of α values, uniformly sampled in the range $[\sigma_0, 5\sigma_0]$ with step $0.1\sigma_0$, and feed the ensemble of all such base clusterings to WEAC, which extracts the final consensus clustering. Here, both cross-camera (hypothesis H_0) and same-camera (hypothesis H_1) correlations are modeled through generalized Gaussian distributions, and their parameters, including σ_0, are estimated through the expectation-maximization algorithm. Since the shape parameter $\beta = 2$ (Gaussian) turned out to be near-optimal, it was selected for the sake of simplicity. Finally, given the weights, we perform Correlation Clustering by the fast greedy algorithm described in [78], based on the binary Markov Random Fields optimization of [88].

2.3.3 Consensus clustering

Once obtained the set of base clusterings $\mathscr{P} = \{P^1, P^2, \ldots, P^M\}$, we run WEAC as described in [69] to obtain the consensus clustering Q. The latter, however, depends also on the number of output clusters, l, that is

$$Q^l = \text{WEAC}(\mathscr{P}, l) \tag{2.24}$$

This parameter can be known in advance, or else must be estimated itself from the data, and several methods have been proposed in the literature [53, 86] to this end. In our case, however, considering again the subsequent refinement step, these methods are not really suitable, because we are not interested in the clustering that best explains the data but rather in any compact conservative clustering that guarantees the absence of wrong fusions. In other words, we are ready to accept false negatives (same-camera residuals that are dispersed over several clusters), but want to avoid false positive (cross-camera residuals included in the same cluster). Therefore, based on experimental evidence, we devised a simple *ad hoc* criterion.

Let us consider the sequence of consensus partitions $\{Q^1, Q^2, \ldots\}$ output by WEAC as the number of output clusters, l, goes from 1 (single cluster) to some suitable maximum, possibly, the number of data points. As l increases, new clusters emerge through the splitting of existing clusters. Initially, such splits separate cross-camera groups of residuals, or same-camera groups with markedly different features. In these cases, since a large number of residuals are involved, the similarity between subsequent partitions Q^{l-1} and Q^l is relatively low. When all these compact groups have been separated, further increases of the parameter correspond typically to the separation of just a few residuals from some existing cluster, with little impact on partition similarity. Therefore we implement a simple test on similarity, and select the parameter l^* as the smallest value of l

beyond which partitions do not change appreciably anymore. In formulas, let ϵ be a small positive value, say, $\epsilon = 10^{-2}$, and

$$s(l) = \mathrm{sim}(Q^{l-1}, Q^{l}) \tag{2.25}$$

Then l^* is defined by the conditions

$$s(l^* - 1) < 1 - \epsilon, \text{ and } s(l) > 1 - \epsilon, \forall l \geq l^* \tag{2.26}$$

With this value, we obtain the desired consensus clustering Q^*.

2.3.4 Refinement step

This is a key part of the proposed method. Relying on the hypothesis that previous steps provided pure clusters, with reliable PRNU estimates, the clusters are progressively merged while better and better PRNU estimates emerge.

Given the consensus clustering $Q^* = \{C_1, C_2, .., C_{l^*}\}$, we estimate the camera PRNU associated with cluster $\mathscr{C}_p$ as the sample mean, $\widetilde{K}_p$, of its members. If the cluster is large enough, the estimate is relatively unaffected by noise, and the correlation between $\widetilde{K}_p$ and a generic residual, R_i, gives a clear indication on whether R_i is cross-camera (H_0) or same-camera (H_1) with respect to $\widetilde{K}_p$. More formally, let

$$\rho_i = \mathrm{corr}(\widetilde{K}, R_i) \tag{2.27}$$

indicate the correlation between a PRNU estimate and a residual, and

$$f_{C_p}(\rho|H_0), \text{ and } f_{C_p}(\rho|H_1) \tag{2.28}$$

be the probability density functions (pdf) of the PRNU-residual correlation for cluster C_p under the hypotheses H_0 and H_1, respectively. Given such

pfds, we also have the log-likelihoods of the two hypothesis for the same residual R_i

$$\begin{cases} L_p(R_i|H_0) = \log f_{C_p}(\rho_i|H_0) \\ L_p(R_i|H_1) = \log f_{C_p}(\rho_i|H_1) \end{cases} \tag{2.29}$$

Therefore, assuming the residuals of a cluster to be independent of one another, given H_0 or H_1, we can compute the log-likelihoods of the two hypotheses with respect to K_p for a whole cluster C_q by summing their individual log-likelihoods, obtaining eventually the decision statistic

$$\Lambda_{pq} = \sum_{R_i \in C_q} [L_p(R_i|H_1) - L_p(R_i|H_0)] \tag{2.30}$$

These statistics will guide our refinement algorithm. To compute them, we must estimate the same-camera and cross-camera distributions for each cluster C_p. Towards this aim, we compute the correlation index between K_p and all residual $R_i \notin C_p$, and run the expectation-maximization (EM) algorithm assuming a generalized Gaussian distribution under both hypotheses. However, it may happen that all same-camera residuals have been already collected in cluster C_p, in which case the EM could not work properly. To obviate this problem we augment the set of same-camera correlations through a leave-one-out approach. That is, we remove one of the residuals, say R_j, from the cluster and compute a new estimate of the cluster PRNU without R_j

$$\widetilde{K}'_{p,j} = \frac{1}{|\mathscr{C}_p| - 1} \sum_{R_i \neq R_j \in \mathscr{C}_p} R_i \tag{2.31}$$

Although $\widetilde{K}'_{p,j}$ does not coincide with $\widetilde{K}_p$, it is a good approximation of it whenever $|C_p|$ is not too small. At this point we can compute the correlation

$$\rho'_j = \mathrm{corr}(\widetilde{K}'_{p,j}, R_j) \tag{2.32}$$

Algorithm 1 Clustering Refinement

procedure $C = \text{REFINEMENT}(\{R\}, \{C\})$
  $T_{|C|} = 0$
  do
    $changed = \text{FALSE}$
    Increase $T_{|C|}$
    $\mathscr{L} = \{C_p : |C_p| > T_{|C|}\}$
    $\mathscr{S} = \{C_p : |C_p| \leq T_{|C|}\}$
    compute $\widetilde{K}_p$ for all $\mathscr{C}_p \in \mathscr{L}$
    compute Λ_{pq} for all $\mathscr{C}_p \in \mathscr{L}$ and $\mathscr{C}_q \in \mathscr{S}$
    while $\max_{(p,q)} \Lambda_{p,q} > 0$ **do**
      $changed = \text{TRUE}$
      $(p^*, q^*) = \arg\max_{(p,q)} \Lambda_{p,q}$
      $C_s = \text{merge}(C_{p^*}, C_{q^*})$
      update $\mathscr{L}, \mathscr{S}$
      update K_p for all $\mathscr{C}_p \in \mathscr{L}$
      update Λ_{pq} for all $\mathscr{C}_p \in \mathscr{L}$ and $\mathscr{C}_q \in \mathscr{S}$
    end while
  while *changed*
end procedure

and repeat it for all $R_j \in C_p$, contributing a total of $|C_p|$ new samples of same-camera correlations. Note that all PRNU-residual correlations can be computed based only on the matrix of inner products as

$$\begin{aligned} \rho_i &= \text{corr}(\widetilde{K}, R_i) = \text{corr}\Big(\frac{1}{|C_p|}\sum_{R_j \in C_p} R_j,\ R_i\Big) \\ &= \frac{\sum_{R_j \in C_p} c_{ji}}{\sqrt{\sum_{R_j \in C_p}\sum_{R_k \in C_p} c_{jk}}\ \sqrt{c_{ii}}} \end{aligned} \tag{2.33}$$

Therefore, we do not really need PRNU estimates to compute correlations, which makes refinement a very fast process.

We can now describe the iterative refinement procedure, also with reference to the pseudo-code shown in Algorithm 1.

The generic step is characterized by a threshold, $T_{|C|}$, which divides the current clusters in two sets

$$\mathscr{L} = \{C_p : |C_p| > T_{|C|}\}, \;\; \mathscr{S} = \{C_p : |C_p| \leq T_{|C|}\} \tag{2.34}$$

namely, the sets of large and small clusters, $\mathscr{L}$ and $\mathscr{S}$. Here, "large" and "small" depend on $T_{|C|}$, which is a running threshold (not a parameter), raised progressively during iterations so as to merge first the smallest clusters and then proceed with larger ones.

At each iterations we try merging each small cluster with one of the large clusters. With this aim, we compute the statistics Λ_{pq}, for each $C_p \in \mathscr{L}$ and all $C_q \in \mathscr{S}$, and sort them in descending order. If the largest Λ_{pq} is positive, the corresponding clusters are merged, then, the statistics for the newly formed cluster are recomputed, all lists are updated, and the algorithm proceeds in the same way until all statistics become negative. At this point $T_{|C|}$ increases, moving the boundary between large and small clusters. That is, some large clusters move from $\mathscr{L}$ to $\mathscr{S}$, and become candidates to be merged with larger ones. Moreover, small clusters that remain isolated in a given step may still be merged in later steps, as PRNU estimates keep improving and statistics change. The algorithm stops when a single cluster remains in $\mathscr{L}$ and no more merging is possible.

2.4 Experimental results

To validate the proposed method, we carried out a number of experiments in a large variety of conditions, comparing performance with the best state-of-the-art references. Besides using the original images as they are output from the cameras, we consider also the more realistic case of

images subject to compression and resizing, as customary on social networks. In the following subsections, we describe the datasets used for the experiments, the performance metric, some implementation details, and finally the experimental results.

2.4.1 Datasets

In order to guarantee full reproducibility of results we use the publicly available Dresden Image Database [44] described in the previous chapter.

For our experiments, we selected 10 models (see Table 2.1), including a minimum of 2 and a maximum of 5 devices per model. The original images are relatively large, from 5 to 12 Mpixels, and uncompressed. Upon uploading on Facebook they are resized and JPEG compressed with size and quality factor depending also on a quality toggle selected by the user [89]. In the low quality (LQ) modality, the size is about 1 Mpixel, and the JPEG quality factor goes from 70 to 90.

We built a number of heterogeneous datasets to explore various situations, more precisely

Set A: models { I70, Z150, D200, μ, RCP };

Set B: models { M1063, S710, DCZ, L74w, NV15 };

Set C: all ten models.

For each dataset (say X) we consider three version, with one (X.1) two (X.2) or all (X.max) devices per model, in order to explore the dependence on the total number of devices in the set. Therefore, the largest set, C.all, includes 39 devices. We designed the disjoint sets A and B because, for some reference techniques, a few key parameters must be estimated on a training set: some thresholds for Bloy2008, the threshold Th for Amerini2014, and the α and β parameters for Marra2016. So, if set A is under test, parameters are estimated on set B and viceversa. For set C we

Make	Model	Acronym	#Devices	#Images	Original	Facebook HQ	Facebook LQ
Canon	Ixus 70	I70	3	567	3072 x 2304	2048 x 1536	1152 x 864
Casio	EX-Z150	Z150	5	924	3264 x 2448	2048 x 1536	1224 x 918
Kodak	M1063	M1063	5	2391	3664 x 2748	2048 x 1536	1374 x 1031
Nikon	CoolPix S710	S710	5	925	4352 x 3264	2048 x 1536	1088 x 816
Nikon	D200	D200	2	752	3872 x 2592	2048 x 1370	968 x 648
Olympus	μ1050SW	μ	5	1040	3648 x 2736	2048 x 1536	1368 x 1026
Pratika	DCZ5.9	DCZ	5	1019	2560 x 1920	2048 x 1536	960 x 720
Rollei	RCP-7325XS	RCP	3	589	3072 x 2304	2048 x 1536	1152 x 864
Samsung	L74wide	L74w	3	686	3072 x 2304	2048 x 1536	1152 x 864
Samsung	NV15	NV15	3	645	3648 x 2736	2048 x 1536	1368 x 1026
Σ	9		39	9538			

Table 2.1. Main features of the dataset used in the experiments, extracted from the Dresden dataset. Original images are uncompressed and full size. Images uploaded and downloaded from Facebook are resized and JPEG compressed.

are forced to perform the estimation on models, or even individual devices, already present in the dataset. Note that no training set is necessary for the proposed method.

To these heterogeneous datasets we add all homogeneous datasets, including all devices of a single camera model. This is an important and challenging test, because residuals might contain model-related micropatterns, due to the internal image processing chain, which boost clustering performance. In homogeneous datasets, such micropatterns coincide for all devices and cannot be exploited.

2.4.2 Results

The proposed algorithm has been developed in Matlab under Linux. To ensure full reproducibility of research, our software is freely available online at www.grip.unina.it. Moreover, the Correlation Clustering source code is available at www.wisdom.weizmann.ac.il/∼bagon/matlab.html, while the WEAC software can be reached through the link https://arxiv.org/abs/1405.1297. The reference methods have been implemented as described in the original papers.

By using Matlab, we renounce some efficiency in favor of simpler development and better readability. This also prevents an accurate assessment of complexity. On the other hand, most of the computational burden is associated with the unavoidable pre-processing phase. The denoising of N images, and the computation of inner products between $N(N-1)/2$ couples of residuals largely dominates the complexity. In fact, the remaining processing tasks do not access the heavy original data anymore, and base all their computations on the matrix of inner products. For our largest dataset, actual clustering (including the generation of multiple CC partitions) took a few dozens seconds, as opposed to about two hours for the pre-processing phase.

Set	#Dev	Bloy08	Amerini14	Fahmy15	Marra16	Marra17	*Ncut oracle*	*CC oracle*	*WSCE true-k*
A.1	5	0.708	0.763	0.707	0.665	**0.916**	*0.872*	*0.908*	*0.832*
A.2	10	0.725	0.699	0.683	0.813	**0.852**	*0.801*	*0.848*	*0.606*
A.max	18	0.689	0.568	0.374	0.398	**0.729**	*0.665*	*0.761*	*0.467*
B.1	5	0.388	0.722	0.324	0.911	**0.915**	*0.722*	*0.736*	*0.813*
B.2	10	0.538	0.606	0.505	0.833	**0.836**	*0.683*	*0.819*	*0.789*
B.max	21	0.464	0.451	0.457	0.860	**0.881**	*0.631*	*0.834*	*0.527*
C.1	10	0.627	0.669	0.703	0.844	**0.865**	*0.669*	*0.836*	*0.696*
C.2	20	0.683	0.607	0.486	0.856	**0.956**	*0.607*	*0.929*	*0.723*
C.max	39	0.598	0.536	0.413	0.686	**0.821**	*0.536*	*0.798*	*0.318*

Table 2.2. Performance on heterogeneous sets.

2.4.2.1 Performance metric

Performance is assessed based on the agreement between algorithm clustering and ground truth clustering, measured by the popular Rand Index [90]. Let us consider a set of n data points, and two alternative partitions $P = \{C_1, C_2, \ldots, C_N\}$ and $P' = \{C'_1, C'_2, \ldots, C'_{N'}\}$ of these points. Then let us consider all $\binom{n}{2}$ couples of points: the two partitions are in good agreement if the majority of couples have the same treatment in the two cases, namely, they are kept together (in the same cluster) in both P and P', or are separated (put in different clusters) in both P and P'. To measure agreement the following counters are kept:

- a_1: couples falling in the same cluster in both partitions;
- a_2: couples falling in different clusters in both partitions;
- d_1: couples falling in the same cluster in P and in different clusters in P';
- d_2: couples falling in different clusters in P and in the same cluster in P'.

Based on these agreement (a_1 and a_2) and disagreement (d_1 and d_2) counters, the Rand Index is defined as

$$RI = \frac{a_1 + a_2}{a_1 + a_2 + d_1 + d_2} = (a_1 + a_2) / \binom{n}{2} \tag{2.35}$$

This index goes from 0 (total disagreement) to 1 (perfect agreement). However, the average Rand Index for two random partitions $\overline{RI}$ is a non-zero number which depends deterministically on the cluster cardinalities. To get rid of this bias, we resort to the Adjusted Rand Index (ARI) [91], defined as

$$ARI = (RI - \overline{RI})/(1 - \overline{RI}) \tag{2.36}$$

The ARI decreases quite fast as the clustering under test departs from the ground truth, therefore, values close to 1 indicate a near-perfect agreement. Imperfect but informative clusterings have a positive ARI, while values close to zero or negative indicate serious issues.

2.4.2.2 Results on original images

Table 2.2 shows experimental results for the selected datasets. Together with the proposed method, we consider some reference methods proposed in latest years (see Section II), shortnamed Bloy2008 [8], Fahmy2015 [64], Amerini2014 [67], and Marra2016 [55]. In addition, as a further reference, we show the results provided by the oracle versions of Ncut clustering [68] and Correlation Clustering [77], with parameters set *a posteriori* so as to maximize performance. Therefore, such results upper bound the performance achievable by these state-of-the-art clustering methods. We also show results for the recently proposed [82] weighted spectral cluster ensemble (WSCE) algorithm, with the number of clusters set to the true number of devices (true-k) assumed known. For each dataset, the best result (excluding oracles) is highlighted in blue.

The proposed method provides the best performance on all datasets, sometimes with a wide margin w.r.t. the second best. As an example, on the largest dataset, C.max, the proposed method has an ARI of 0.821 as opposed to the 0.686 of Marra2016, the second best. In addition, it performs always better than both oracles and WSCE, with the only exception of dataset A.max where CC oracle is slightly better. These results fully support our approach. In particular, the consistent performance gain w.r.t. CC oracle underlines the importance of the refinement phase, which exploits the gradually improving PRNU estimates to make some critical decisions. On the other hand, the improvements w.r.t. Marra2016 speak in favor of other innovative design choices made in this work, notably, the use of consensus clustering, and the adoption of a model-based decision

Set	#Dev	Bloy08	Amerini14	Fahmy15	Marra16	Marra17	*Ncut oracle*	*CC oracle*	*WSCE true-k*
I70	3	0.852	0.567	0.599	0.891	**1.000**	*1.000*	*0.958*	*0.995*
Z150	5	0.421	0.205	0.386	**0.566**	0.471	*0.432*	*0.540*	*0.059*
M1063	5	0.662	0.421	0.247	0.525	**0.781**	*0.692*	*0.681*	*0.545*
S710	5	0.025	0.485	0.531	0.756	**0.988**	*1.000*	*0.968*	*0.782*
D200	2	0.305	**1.000**	0.605	0.833	**1.000**	*1.000*	*0.949*	*1.000*
μ	5	0.460	0.689	0.495	**0.948**	0.789	*0.762*	*0.776*	*0.642*
DCZ	5	0.681	0.008	0.283	0.787	**0.963**	*0.958*	*0.980*	*0.997*
RCP	3	0.778	**1.000**	0.271	0.953	0.954	*1.000*	*0.911*	*0.979*
L74w	3	0.687	0.924	0.374	0.851	**1.000**	*1.000*	*0.980*	*1.000*
NV15	3	0.808	**1.000**	0.457	**1.000**	0.988	*1.000*	*0.930*	*0.567*
average		0.568	0.630	0.425	0.811	**0.893**	*0.884*	*0.867*	*0.757*

Table 2.3. Performance on homogeneous sets. For each model all available devices and images are used.

statistic. It is worth reminding that, contrary to most reference techniques, the proposed algorithm does not need user-defined parameters nor estimates them on external training sets. As expected, the performance generally impairs, although not monotonically, as the number of devices in the dataset increases. Nonetheless, even on the 39-device C.max, the proposed method provides a very accurate clustering. Table 2.3 shows experimental results for the homogeneous datasets, comprising several devices of the same model. Contrary to our expectation, there is no performance impairment, on the average, with results that are basically aligned with those observed for the smallest heterogeneous datasets. Instead, the dependence on the number of devices is fully confirmed. The performance is very good for datasets with just 2 or 3 devices, with Ncut oracle providing always perfect clustering, while it impairs for 5-device datasets. A negative peak is observed for the Casio EX-Z150 model, due to some artifacts in the PRNU, already noticed in [92] and [76]. On these homogeneous datasets the proposed method is not always the best, but keeps providing the best average performance, with a significant lead over the second best, Marra2016. Again, its performance is slightly *better* than both oracles and WSCE.

2.4.2.3 Results on images uploaded from Facebook

With Table 2.4 we begin investigating the behavior of the proposed and reference methods in the presence of image impairments. To this end, all images have been uploaded on Facebook, selecting the high quality (HQ) modality, and then downloaded again. Facebook automatically resizes and compresses the images, but the HQ option guarantees that only minor impairments are present. Even so, by comparing results with those of Table 2.2, a significant performance impairment is observed for all methods, with a loss of 0.3 points on the average. Some methods seem to suffer more than others the lower image quality, notably, Fahmy2015 and Amerini2014, to-

Set	#Dev	Bloy08	Amerini14	Fahmy15	Marra16	Marra17	*Ncut oracle*	*CC oracle*	*WSCE true-k*
A.1	5	0.272	0.301	0.196	0.402	**0.723**	*0.305*	*0.513*	*0.198*
A.2	10	0.395	0.233	0.106	0.471	**0.592**	*0.294*	*0.567*	*0.141*
A.max	18	0.244	0.134	0.033	**0.538**	0.532	*0.171*	*0.569*	*0.096*
B.1	5	0.529	0.592	0.098	0.691	**0.847**	*0.705*	*0.792*	*0.433*
B.2	10	0.355	0.357	0.036	0.621	**0.718**	*0.394*	*0.683*	*0.226*
B.max	21	0.232	0.078	0.019	0.547	**0.644**	*0.084*	*0.597*	*0.186*
C.1	10	0.461	0.309	0.089	0.618	**0.646**	*0.338*	*0.643*	*0.323*
C.2	20	0.324	0.173	0.064	**0.573**	0.485	*0.203*	*0.636*	*0.111*
C.max	39	0.220	0.031	0.026	0.571	**0.601**	*0.049*	*0.589*	*0.240*

Table 2.4. Performance on heterogeneous sets after high quality uploading on Facebook.

Set	#Dev	Bloy08	Amerini14	Fahmy15	Marra16	Marra17	*Ncut oracle*	*CC oracle*	*WSCE true-k*
A.1	5	0.001	0.004	0.007	**0.115**	0.104	*0.007*	*0.162*	*0.020*
A.2	10	0.001	0.033	0.063	0.052	**0.323**	*0.059*	*0.324*	*0.013*
A.max	18	0.002	0.015	0.008	0.034	**0.210**	*0.017*	*0.182*	*0.013*
B.1	5	0.002	0.010	0.019	**0.414**	0.204	*0.140*	*0.229*	*0.192*
B.2	10	-0.002	0.464	0.102	0.109	**0.670**	*0.528*	*0.640*	*0.034*
B.max	21	0.000	0.264	0.025	0.151	**0.327**	*0.268*	*0.392*	*0.055*
C.1	10	0.001	0.010	0.016	0.063	**0.296**	*0.022*	*0.328*	*0.002*
C.2	20	0.004	0.041	0.001	0.062	**0.176**	*0.041*	*0.219*	*0.016*
C.max	39	0.001	0.018	0.003	0.055	**0.293**	*0.023*	*0.237*	*0.031*

Table 2.5. Performance on heterogeneous sets after low quality uploading on Facebook.

gether with the Ncut oracle, and especially WSCE. This latter method is probably penalized by the constraint on the number of clusters which, with very noisy data, leads to highly impure clusters. The CC oracle, together with the CC-based methods, Marra2016 and the current proposal, show smaller losses, suggesting this graph partitioning approach to be more conservative than Ncut. The proposed method keeps being the best performer on 7 datasets out of 9, losing to Marra2016 in two cases. When the low quality option is used, Table 2.5, the single-image PRNU estimate represented by image residuals becomes quite unreliable and, accordingly, the performance decreases drastically, to the point that the majority of methods become basically useless. Also the proposed method suffers from such low quality inputs. It keeps being the best on 7 out of 9 datasets, but the performance impairs significantly, and the corresponding clusterings, though meaningful (positive ARI) are not much reliable. Arguably, in such conditions, one cannot rely only on the PRNU to make decisions, and further information should be collected to complement the image residuals.

2.4.2.4 Alternative consensus clustering tools

In the proposed method, we use a specific consensus clustering algorithm, WEAC, but many more have been proposed in the literature [54]. Testing all such methods is out of the scope of this work, but we tried to replace WEAC with the hybrid bipartite graph formulation (HBGF) proposed in [81], which fits nicely in the whole algorithm, and whose code is published online by the authors. The results, summarized in Table 2.6, show WEAC to be clearly preferable, and especially more robust than HBGF, with a strong performance gain especially on the larger datasets.

Set	original images		facebook HQ	
	WEAC	HBGF	WEAC	HBGF
A.1	0.916	0.397	0.723	0.602
A.2	0.852	0.665	0.592	0.541
A.max	0.729	0.283	0.532	0.471
B.1	0.915	0.677	0.847	0.791
B.2	0.836	0.786	0.718	0.660
B.max	0.881	0.822	0.644	0.436
C.1	0.865	0.532	0.646	0.378
C.2	0.956	0.807	0.485	0.537
C.max	0.821	0.347	0.601	0.386

Table 2.6. Comparison among ensemble clustering tools.

2.4.2.5 Visual inspection

To gain a deeper insight on the algorithms' behavior, Fig.2.6 and Fig.2.7 provides a graphical representation of some sample results, as already done in [67]. To save space, we show results only for set B.1, with both original images (Fig.2.6) and Facebook high-quality images (Fig.2.7). Each bar-graph shows (up to 18) clusters retrieved by the methods under comparison: Bloy2008, Amerini2014, Fahmy2015, Marra2016, proposed. The total number of retrieved clusters and the ARI measure are shown top-right. Each bar may show different colors, proportional to the number of cameras in the cluster that come from each of the five devices. The legend on the bottom shows, for each device, the associated color, the corresponding camera model, and number of images in the dataset. With both original and facebook datasets, the proposed method provides large and pure clusters for all cameras. Using a small set (5 devices) allows one to gain full insight on performance, a difficult task when many more clusters are involved. Marra2016 and the proposed method provide the best results, as already clear from the ARI figures. In both cases,

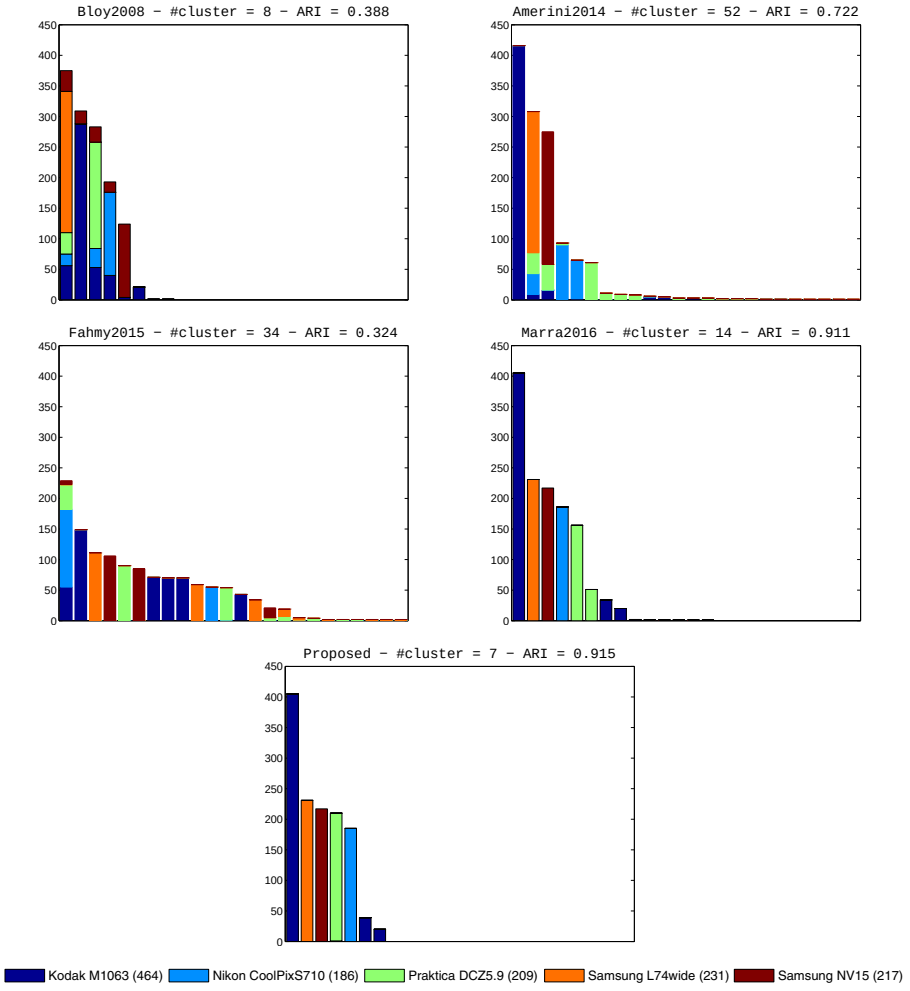


Figure 2.6. Graphical representation of clustering results on the B.1 dataset with original images.

all detected clusters are pure or almost pure and there are large clusters associated with each camera which allow a reliable estimation of the corresponding PRNU pattern. The only wrong decisions concern the separation of a few small clusters from the main ones. This happens for the Kodak

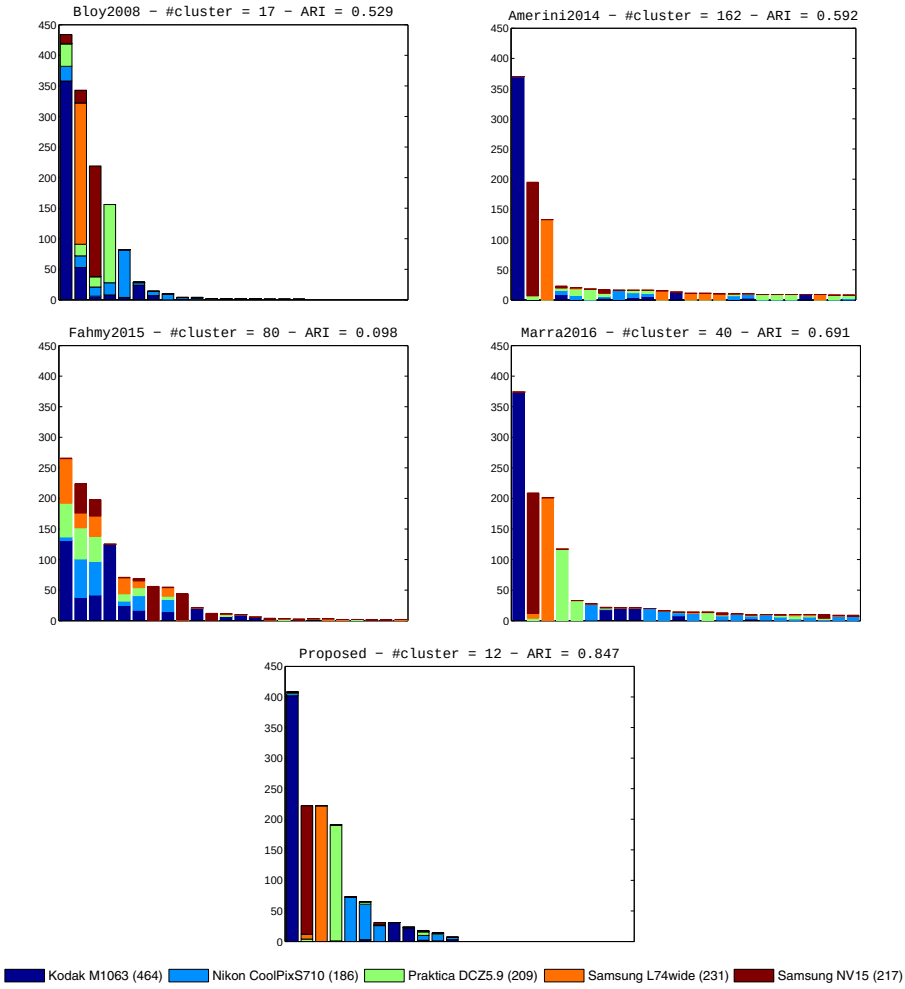


Figure 2.7. Graphical representation of clustering results on the B.1 dataset with HQ facebook images.

and the Praktica devices, with Marra2016, and only for the Kodak with the proposed method. On the contrary, all other methods generate some mixed clusters, which lead to inaccurate estimates of the PRNU pattern. Moreover, Fahmy2015 is affected by a strong cluster fragmentation, while

Amerini2014 produces a large number of singletons and very small clusters. The performance impairs significantly for all methods when the images are downloaded from Facebook. Even Marra2016 generates only 4 large clusters, now, with images of the Nikon camera dispersed over many small clusters. In this case, despite the impairment, the proposed method provides clearly the best result, with four near-perfect clusters and the other ones, two of which relatively large, accounting mostly for the Nikon images.

Chapter 3

Forgery localization in a blind scenario

The wide diffusion of powerful image editing tools has made image manipulation very easy. This impacts on many fields of life, and is especially dangerous in the forensic field, where images may be used as crucial evidence in court. Therefore, in the last decade, digital image forensics has grown tremendously, and new methodologies have been developed to track an image source and determine its integrity. In particular, the interest has focused on passive techniques, which detect traces of manipulations from the analysis of the image itself, with no need of collaboration on the part of the user.

Some of the most successful camera-based methods rely on the PRNU. Its use was first proposed in [2], both for source identification and forgery localization. In this section we focus on PRNU-based methods for forgery detection and localization.

Several improvements have been proposed with respect to the basic method of [2]. in [4] a predictor is estimated which adapts the statistical

decision test locally to take into account image features, such as texture, flatness and intensity, thus reducing the probability of false alarms. In [74], instead, the problem is recast in terms of Bayesian estimation, using a Markov random field (MRF) prior to model the strong spatial dependencies and take decisions jointly rather than individually for each pixel. In [93] and [75] the problem of small forgery detection is addressed, resorting to image segmentation and guided filtering to improve the decision statistics. Further improvements have been recently proposed by considering the use of discriminative random fields [94] or by introducing multiscale analysis [95].

All these methods rely on the assumption that a large number of images are available, which are known to come from the camera of interest. However, such an hypothesis is not reasonable in a real-world scenario. Therefore, in this paper we propose and analyze a framework for image forgery localization in a blind scenario [73]. We only assume to have a certain number of images, whose origin, however, is unknown. Then we estimate one or more PRNU's by means of a blind source clustering algorithm and use them to establish the integrity of the image under test. In the following Section we describe the PRNU-based framework for blind forgery localization, while in Section 3 present experimental results of [96] with reference to various clustering approaches [8, 67, 55].

3.1 Camera-based Forgery Localization Framework

In both camera identification and forgery localization tasks, the PRNU of the camera of interest is given in advance, or is accurately estimated from a large number of images coming from the camera. However, in many forensic scenarios, and especially in investigation, no information is available on the origin of the images under analysis, neither the probe nor the dataset. Often, however, it is reasonable to assume that the images

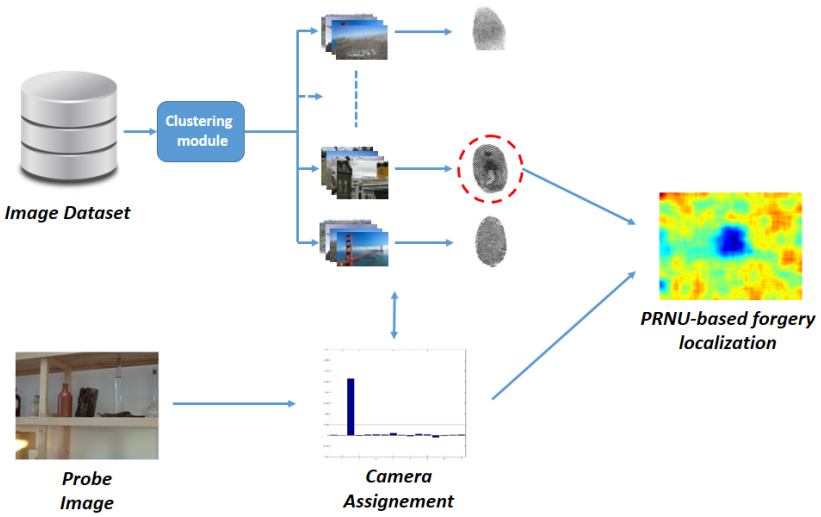


Figure 3.1. A framework for PRNU-based forgery localization in a blind scenario.

in the dataset come from just a few different devices. With this assumption, we can pursue PRNU-based forgery localization in a blind scenario, following the framework shown in Fig.3.1 and already outlined in [73].

The proposed framework consists of four steps

1. Residual-based image clustering;
2. Cluster PRNU estimation;
3. Camera assignment;
4. Forgery localization.

The first two steps allow us to group together images coming from the same camera and to estimate their PRNU. Then, in step 3, the test image is associated with one of the clusters (or possibly none) by a PRNU-based correlation test. Finally, the tampered area of the test image is localized

by detecting the absence of the selected PRNU. These steps are described in more detail in the following.

3.1.1 Residual-based image clustering

To perform PRNU-based forgery localization one needs the true PRNU of the camera. Otherwise, it can be estimated by averaging a large number of images taken by the camera of interest. To this end, the first step of the proposed framework aims at grouping together all images of the dataset coming from the same camera. Since these share the same PRNU, they will exhibit a larger correlation than images coming from different cameras. However, before computing correlations, the high-level content of the images, which represents an interference in this context, is removed by high-pass filtering, obtaining the so-called noise residuals.

Let $\mathscr{R} = R_1, R_2, \ldots, R_N$ be the set of all noise residuals in the dataset. We want to partition this set in M distinct clusters, where the number of clusters is not know a priori. Therefore, the output of this step is a partition, P, of the dataset, namely:

$$P = \{C_1, C_2, ..., C_M\} \qquad C_i \cap C_j = \emptyset \; \forall i \neq j, \qquad \bigcup_{i=1}^{M} C_i = \mathscr{R} \tag{3.1}$$

In the literature, a number of PRNU-based clustering methods have been proposed recently [56, 67, 8, 9], some of which will be considered in the experiments. Ideally, we would like to obtain as many clusters as are the source devices in the dataset, $M = M_t$, with M_t the number of devices, and all of them "pure", namely consisting only of images taken by the same device. In practice, the estimated number of clusters may differ from the number of cameras and, even when they coincide, the clusters may not be pure, comprising images coming from different sources. In all cases, the effect is a loss of accuracy in PRNU estimation. When under-

partitioning occurs, $M < M_t$, clusters are necessarily "impure", comprising also images coming from other cameras which act as additional noise in the estimation. In case of over-partitioning, $M > M_t$, even pure clusters may comprise only a fraction of all images taken by a camera, leading to a less reliable estimate. The aforementioned effects may both show up in the same clustering experiment. Of course, all deviations from perfect clustering tend to cause a loss of performance

3.1.2 Camera fingerprint estimation

In the second step, each cluster is treated as "pure", and used to estimate both the PRNU and the predictor needed in the localization phase [4]. Given N_m images in the m-th cluster, one can perform a maximum-likelihood (ML) estimate of the PRNU as [4]

$$\widehat{K}_m = \sum_{i=1}^{N_m} \left[\frac{I_i}{\sum_{i=1}^{N^m} I_i^2} \right] R_i \tag{3.2}$$

In alternative, one can use the simpler sample average

$$\widehat{K}_m = \frac{1}{N_m} \sum_{i=1}^{N_m} R_i \tag{3.3}$$

which ensures very close performance to the ML case, provided N_m is large enough. On the other hand, when the cluster is too small, both estimates become quite unreliable because the noise residuals, R_i, have a very small signal component overwhelmed by noise. Whatever the estimator, some suitable steps follow to remove non-unique artifacts originated by other camera processes.

Some clustering methods tend to generate a large number of small clusters, even singletons, besides a few large ones. It makes sense to discard such small clusters, due to the ensuing unreliable estimates. Therefore, we

introduce a parameter, $N_{\min}$, left to the analyst to set, such that all clusters with $N_m < N_{\min}$ are automatically discarded, avoiding their involvement in the forgery localization process.

Besides the PRNU itself, the localization algorithm proposed in[4] needs a predictor, which establishes the expected value of the correlation for a pristine image. Therefore, for each cluster, we must also estimate the predictor parameters, say Θ_m. To this end, the cluster must be further divided in two subsets, $C_m = C'_m \cup C''_m$. The first one, C'_m, is used to compute an *internal* PRNU, to which images of the second set, C''_m, are correlated. The parameters of the predictor, Θ_m, are then designed to minimize the error between the predicted and observed values of the correlation. Clearly, this further partition of the cluster further stresses the need for it to be large enough. To reduce this problem, we split clusters exactly in half for this task. Note, however, that the final estimate of the cluster PRNU can be carried out from the whole set. Indeed, the test image is completely alien to the cluster, and hence there is no reason to penalize the estimation of the PRNU. In conclusion, the output of this step is the set of estimated PRNUs and predictor parameters, $\{\widehat{K}_m, \Theta_m, m = 1, \ldots, M\}$.

3.1.3 Camera assignment

In this step we try to establish whether the probe image, I_p, is compatible with one of the estimated PRNU's, and which one. This decision is based on the normalized correlation[1]

$$\rho_m = \mathrm{corr}(R_p, I_p \times \widehat{K}_m) \tag{3.4}$$

between the high-pass image residual, R_p, and each of the scaled fingerprints.

[1] Here, and throughout this work, we assume the images to be perfectly aligned. Otherwise, one can replace normalized correlation with Peak-to-Correlation Energy (PCE) ratio [97], which works correctly also in the presence of image cropping.

The probe image is assumed to come from the camera with the most correlated PRNU

$$\widehat{K}_{\max} = \arg\max_{m} \operatorname{corr}(R_p, I_p \times \widehat{K}_m) \tag{3.5}$$

which is therefore selected to perform forgery localization. However, it is also possible that none of the cameras under analysis originated the probe image, in which case all correlations should be small. To formalize this problem, let us consider the two hypotheses

H_0 : the probe image is alien to the dataset
H_1 : the probe image comes from one of the dataset cameras

To design a statistical test we should know the distribution of ρ under both hypotheses. This is not possible in our blind scenario, therefore we resort to a Neyman-Pearson test, selecting a decision threshold, t, which guarantees a suitably small false alarm probability P_{FA}. Following [98], we assume the normalized correlations to have a Gaussian distribution under H_0

$$\rho \sim N(0, 1/HW) \tag{3.6}$$

where H and W are the image dimensions. Therefore

$$\begin{aligned} P_{FA} &= \Pr(\rho_{\max} > t|H_0) = 1 - (1 - \Pr(\rho_m \leq t|H_0))^M \\ &= 1 - (1 - Q(t\sqrt{HW}))^M \simeq MQ(t\sqrt{HW}) \end{aligned} \tag{3.7}$$

with the latter approximation holding for small M and $Q(t\sqrt{HW}) \ll 1$. By inverting the above relation the desired threshold is obtained.

3.1.4 PRNU-based forgery localization

In the last step of the framework, a PRNU-based forgery localization technique is applied. Several such methods have been proposed in the last few years, and they all share the same basic idea. When the image is tampered with, for example through the splicing of some alien material, its PRNU is locally removed. Therefore, a sliding-window correlation test is performed, and when the local correlation index falls below a given threshold, a forgery is declared. Since the correlation may also depend on the image content, the threshold must be adapted locally by using the predictor with parameters $\Theta_{\max}$ estimated in step 2.

The output of this localization step is a binary decision mask that highlights the pixels that are considered as tampered. Given such a mask, and the corresponding ground truth mask, one can compute a number of performance indicators. However, it is worth pointing out that the output mask should be always analyzed by a human interpreter. In fact, real-life image forgeries are performed with a purpose, and they possess a semantics that is not easily captured by algorithms. The localization mask should be therefore regarded as a diagnostic tool to support the expert decision.

3.2 Experimental results

In this section we evaluate the performance of the proposed PRNU-based framework for blind forgery localization. Experiments are carried out on six cameras: Canon EOS-10D, Canon EOS-450D, Canon Ixus 95IS, Nikon D200, Nicon Coolpix S5100, Sony DSC S780. For each camera we use 50 images as training set to perform the PRNU-based clustering and to estimate the cluster PRNUs. Performance is assessed on 50 more images per camera, different from those of the training set. All images have the same size of 768×1024 pixels, and are cropped from the same region of the full-size images. To study forgery localization, we generate forged

Set	NCut-oracle			PCE-PNN			Marra2016		
	ARI	TPR	FPR	ARI	TPR	FPR	ARI	TPR	FPR
Original	0.872	84.31	1.21	0.839	75.74	0.00	0.960	94.79	0.26
JPEG (QF=90)	0.647	61.07	2.77	0.819	79.02	1.50	0.921	93.58	1.33

Table 3.1. Performance of clustering algorithms.

versions of the test images by pasting on them, at the center, a square region of 128×128 or 256×256 sampled randomly from another image. In addition, we repeat the experiments using JPEG compressed images with a quality factor of 90. All the noise residuals are extracted by using the BM3D denoising filter [87], and removing non-unique artifacts caused by demosaicing and lens distortions as proposed in [4]

Localization results are given in terms of ROC curves, giving pixel-wise probability of detection, P_D, and probability of false alarm, P_{FA}, as a function of the decision threshold. As a synthetic measure, the area under the ROC curve (AUC) is also computed. Before considering localization, however, we study the performance of previous steps, to understand their impact on the accuracy.

3.2.1 Image clustering and PRNU estimation

We implemented three clustering algorithms, based on Normalized Cuts [67], on pairwise nearest neighbor (PNN) clustering [8, 73], and on correlation clustering [55], called Marra2016. Note that Ncut requires a threshold parameter to be estimated on a training set, so we consider here an *oracle* version, selecting *a posteriori* the best parameter. For PCE-PNN we used the threshold used by the authors in the original paper. Other PRNU-based clustering methods [8, 58] are not considered here because they have been shown in [67] and [55] to provide a generally worse performance.

Tab.3.1 shows results of clustering algorithms on both original and JPEG compressed images in terms of adjusted rand index (ARI), true positive rate (TPR) and false positive rate (FPR). Marra2016 provides clearly the best results, even better than the oracle version of Ncut, with ARI always very close to 1 (perfect clustering).

In Fig.3.2 we show a graphical representation of results. For uncompressed images (left) Marra2016 provides near-perfect results, with just a few extra clusters for the Sony camera, removed because too small ($N_m < N_{\min}$). In this condition, almost all available images can be used to estimate the PRNU's. The other methods show a higher fragmentation, but clusters are large and pure enough to provide good estimations. Using JPEG compressed images, performance impairs for all methods, but only slightlly so for Marra2016. On the contrary PCE-PNN and Ncut-oracle suffer more on this dataset, especially for the Nikon D200 images, that will not allow a good PRNU estimate.

3.2.1.1 Image to cluster assignment

After clustering the images and estimating the cluster fingerprints, the probe image is correlated we all PRNU's. If the maximum correlation exceeds the decision threshold, t, forgery localization is performed. Together with the 600 test images coming from the selected cameras, we use 600 (negative) images taken from other sources, and cropped to the same size. Tab.3.2 shows the detection performance for a threshold, t, set so as to obtain a theoretical false alarm probability $P_{FA} = 10^{-3}$. In detail, the FPR is the fraction of negative images that pass the test, while the TPR is the fraction of positive images (taken by one of the cameras in the dataset) recognized as such. The FPR is always very small, compatible with the theoretical level. The TPR is also quite large, but almost 6% of the positives are rejected, a fraction that grows above 10% with JPEG compressed images (almost 20% for PCE-PNN).

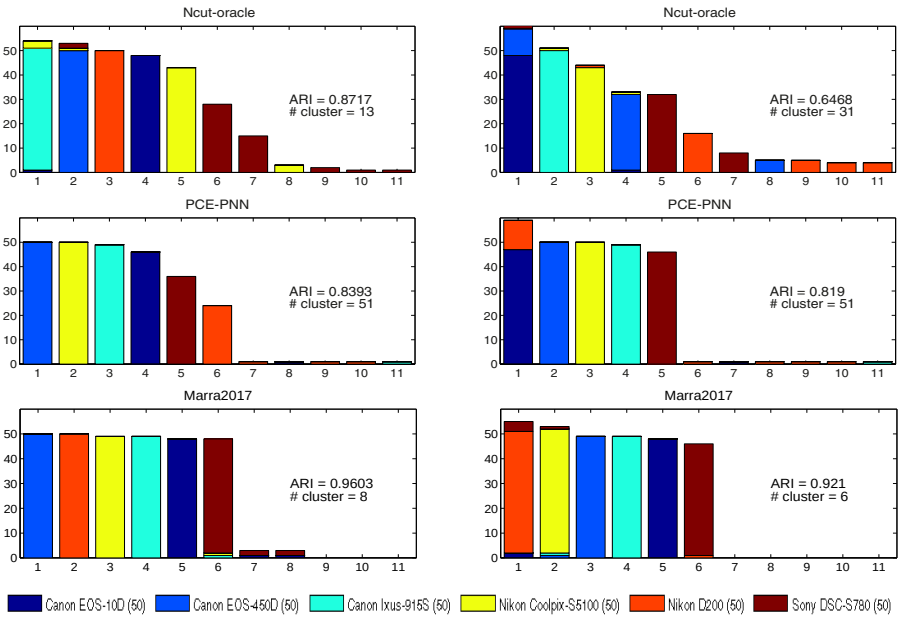


Figure 3.2. Clustering results on original images (left) and JPEG compressed images (right) for Ncut-oracle, PNN-PCE and Marra2016. Colors refer to the devices (see legend) while bar height indicate number of images in a cluster.

Set	Original		JPEG (Qf=90)	
	TPR	FPR	TPR	FPR
Ncut-oracle	94.3%	0%	89.2%	0%
PCE-PNN	94.0%	0.3%	81.0%	0.7%
Marra2016	93.9%	0%	89.6%	1.5%

Table 3.2. Detection performance on original and JPEG compressed images.

Considering that Marra2016 provides near-perfect clustering, these errors must be attributed to the intrinsic problems of PRNU estimation. After correct detection, we could still have a wrong assignment, that is, the

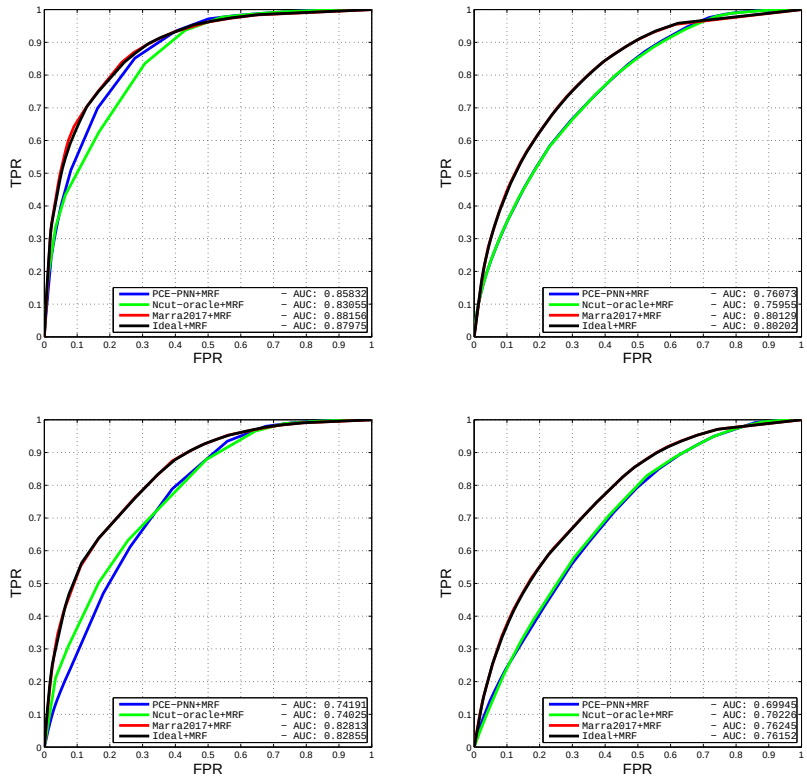


Figure 3.3. Forgery localization results on original (top) and JPEG compressed images (down) with forgeries of 256×256 (left), and 128×128 pixels (right).

probe image could be associated with a wrong camera/PRNU. However, our experiments show this event to be extremely unlikely, with probabilities lower than 0.1% in all cases and not reported in detail for the sake of brevity.

3.2.1.2 Forgery localization

We conclude this analysis by studying forgery localization performance. Localization is carried out by the algorithm proposed in [74], based on a

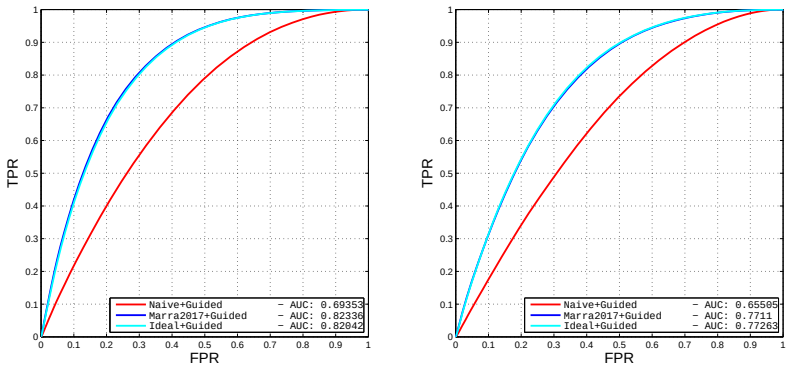


Figure 3.4. Results for clustering-based and "naive" solutions on original (left) and JPEG compressed images (right) with 256×256 pixel forgeries.

MRF prior and on the predictor of [4]. Together with the ideal case where the PRNU's are estimated from all available images, the case of real-world imperfect clustering is also considered, with all methods discussed before. Fig.3.3 shows the ROC curves for original (top) and JPEG compressed images (down) with the two different forgery sizes. With large forgeries on uncompressed images results are very good. The AUC's are close to 0.9 with both ideal and Marra2016 clustering, and only slightly smaller with the other clustering methods.

Surprisingly, Marra2016 provides even a small improvement with respect to ideal clustering,maybe because the discarded images are outliers that impact negatively on the PRNU estimation. As expected, all results impair somewhat when considering smaller forgeries and JPEG compressed images. However, the performance obtained with blind clustering keep being very close (equal for Marra2016) to those of ideal clustering. Finally, we assess the performance when we renounce clustering altogether, computing a single PRNU estimated by averaging all images in the dataset. This "naive" approach makes sense, since the estimated PRNU will bear traces of all camera fingerprints, although attenuated due to the large

number of unrelated images averaged together. Fig.3.4 shows a significant performance drop with respect to the clustering-based solution , both with original and JPEG compressed images (only 256×256 pixel forgeries, for brevity) which fully supports our investigation.

Chapter 4

Feature-based counter forensics

Images and videos are pervasive in all aspects of the modern world, and are becoming ever more important also in forensics, where much evidence in court are based on visual information recorded on digital media. As a consequence, an arms race has long started, as in many other fields related with forensics, between attackers, aiming at falsifying visual evidence for malicious purposes, and defenders who try to reveal possible tampering. Image counter-forensics is of great importance for both sides, as attackers want to conceal traces of their manipulation, and defenders try to remain one step ahead, by exposing themselves possible evasion tools. Among the most frequent and dangerous forms of image manipulation is the insertion of new objects in a photo or, differing only under a semantic point of view, the occlusion of existing objects. Fig.1 shows some examples of image forgeries, crafted with an increasing level of skill on the part of the attacker. Today's editing tools, like PhotoShop or GIMP, besides allowing the easy production of a forgery, include a whole array of tools to conceal their traces, like boundary smoothing, amounting to a basic form of counter-forensics.

The recent trend in image forgery detection is towards the analysis of

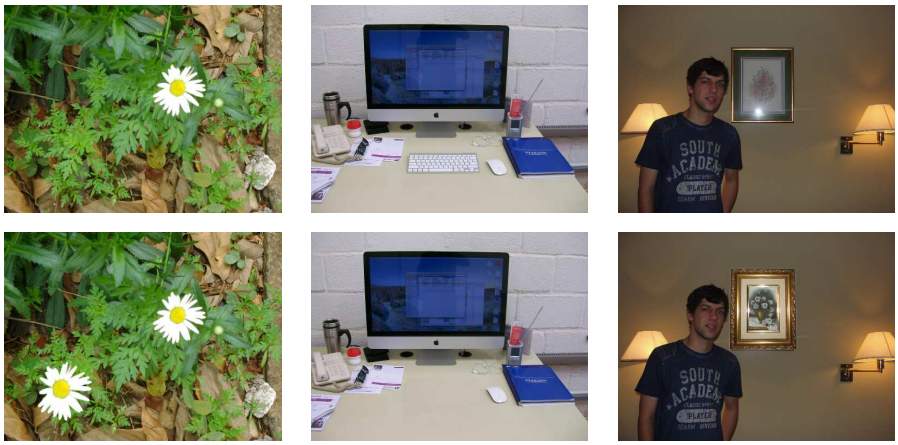


Figure 4.1. Examples of forged images: the copy-moved flower in the left image can be easily spotted by visual inspection; in the center image, the keyboard has been hidden by pieces of background, but there is a suspect change of shade; the frame replaced in the left image cannot be detected without some statistical analysis tools.

image micro-patterns, that is, of the statistical behaviour of the image in small local areas captured by synthetic features (local descriptors). Often, the analysis is carried out not on the original image, but on some high-pass residual, obtained through suitable filtering, since traces of tampering may be found more easily once the low-pass image content has been removed. Machine learning detectors based on local descriptors (LD) provide an extremely promising performance and, quite remarkably, the teams ranking first and second in the first IEEE Image Forensics Challenge on image forgery detection (http://ifc.recod.ic.unicamp.br) both used techniques of this kind. Besides providing good results, techniques based on local descriptors, hence on higher-order statistics, are more resilient to counter-forensic attacks, since the relation between image and features is typically non-invertible.

In this chapter, we propose and analyze some counter-forensic attacks

to LD-based forgery detection techniques presented also in [99]. More in detail, we consider a state-of-the-art detector based on the computation of a histogram of co-occurrences on the image residual [10] and a simpler approach based on Local Binary Pattern (LBP) descriptor [32], proposed originally for texture analysis. The classifier is trained on a suitable training set, comprising splicings (with no specific processing) of various sizes, including small ones. Inspired by [100] we consider both a perfect-knowledge scenario, in which the attacker has full knowledge of the detector, and a limited-knowledge one, where no side information is available. In both scenarios, the proposed techniques perform feature-histogram restoration, with a constraint on image distortion. In our case, however, gradient-descent algorithms cannot be used, and an ad hoc greedy optimization algorithm is therefore proposed. Experimental analysis on a suitable test set proves the effectiveness of these attacks, with an obvious performance gap in favor of the perfect-knowledge scenario.

4.1 Related Work

Counter-forensics is a relatively new topic in the context of image forgery detection [5], especially if compared with more mature fields, like biometrics, digital watermarking, steganography, network security, etc. Kirchner and Böhme have been probably the first researchers to deal with image counter-forensics showing [101] how some rather simple attacks can destroy traces of resampling and casting therefore serious doubts on the effectiveness of some image forensic schemes based on these features. Likewise, Stamm [102] showed that the quantization artifacts caused by JPEG compression can be hidden by adding a dithering noise on the DCT coefficients, restoring approximately the histogram of the original image. Interestingly, it was later observed [103] that dithering introduces its own artifacts, that other forensic tools can discover. This is a rather general

phenomenon, observed also with reference to techniques based on camera sensor noise where the attack [104] is countered by an *ad hoc* method [105] which can be attacked in its turn [106]. Indeed, such two-party problems should be handled through a game-theoretic approach, as proposed in some recent papers [107, 108, 109].

The theoretical results found therein, however, hold only in very special conditions, not met in most practical applications. Notice that changes on isolated image samples or transform coefficients impact on distortion in a simple way, allowing for an easy enforcement of the distortion constraint. When working on more elaborate features this is often not the case. A relatively common approach to counter-forensics is histogram restoration. Several typical image processing tasks, such as contrast-enhancement or gamma correction, but also resampling and compression, leave traces in the image histogram that can be exploited to detect the tampering. Of course, by restoring the original histogram, detection becomes impossible. A possible approach [110] consists in modifying iteratively the histogram, through small changes in the image, making it as close as possible to the histogram of a pristine image, and having care, at the same time, not to degrade visibly the forged image. A similar technique has been subsequently proposed [111] and applied to attack a double-JPEG compression detector. Other detectors tailored to specific features, like color filter array (CFA) artefacts, camera sensor noise and JPEG compression traces, have also been attacked [112, 113] by means of direct injection, using feature decomposition to limit complexity. Notice that changes on isolated image samples or transform coefficients impact on distortion in a simple way, allowing for an easy enforcement of the distortion constraint. When working on more elaborate features this is often not the case.

As a matter of fact, the most promising forgery detectors proposed in the recent literature [114, 115, 116, 9] exploit higher-order statistics, resorting to local descriptors, features computed on a neighborhood of the

target pixels and working also on a pre-filtered high-pass version of the image. Both choices cause the feature associated with a pixel to depend on the mutual relationship among groups of pixels. A notable example is the rich-model descriptors [10], used originally for steganalysis, and found to be very effective also for forgery detection [117] and localization [40, 118, 119]. Since no hypotheses are made on the image tampering process, machine learning detectors based on such features are typically more general and robust than the previous generation. Moreover, with such features, counter-forensics based on histogram restoration becomes much more difucult. In fact, any change in the feature space impacts on a whole group of pixels in the image space, hence on higher-order statistics, and the relationship is typically non-linear, since quantization is part of the feature generation process.

Statistical restoration has been long studied in steganalysis, with both established theoretical results [120] and practical techniques [121]. In that context, the higher-order restoration problem has been also considered by Sarkar [122], for a detector based on second-order dependencies in the DCT domain, using the earth-mover's distance (EMD) formulation, and providing the optimum way of redistributing weights for restoration. Since the problem has a high computational complexity, an heuristic algorithm is proposed for actual implementation. To the best of our knowledge, the only attempt to attack a LD-based image forgery detector is in a 2012 paper [123] where histogram restoration is performed in the feature space to attack Shi's technique [114].

Since we focus, here, on machine learning methods, it is also worth considering adversarial machine learning (also known as adversarial pattern recognition) where similar problems are encountered [124, 125], although not related to image processing, in general, and forgery detection in particular. Especially relevant is the work of Biggio [100] dealing with the detection of counterfeited pdf files, where a problem similar to histogram

restoration is solved by a gradient descent method. However, rather than aiming at a specific feature associated with a genuine item, a more convenient feature is synthesized and pursued by the proposed algorithm, which lies in the acceptance region of the detector just past the decision boundary. This expedient allows for a much faster solution of the problem, although some constraints are necessary to guarantee that the synthetic feature corresponds to a valid pdf file. An attack against a classifier based on Bag-of-Words has been recently used for image classification [126], considering both sparse and dense features, showing that it is possible to modifiy an image without affecting its quality and fooling the classifier. The attack consists essentially in replacing a certain number of selected patches with similar ones belonging to a large dictionary of candidates. Finally, it is worth mentioning an experiment carried out by Nguyen et al. [127], where a state-of-the-art convolutional neural network is induced to classify as "*lion*", with 99.99% confidence, a properly mastered white noise field, shading light on how much room remains for research in this field.

4.2 Forgery Detection Counter-Forensics

Let $X \in \mathscr{X}$ be a pristine image, with $\mathscr{X}$ the image space, for example $\{0, \ldots, 255\}^N$, for N-pixel gray-scale images with 8-bit precision, and $X^0 \in \mathscr{X}$ its forged version. Though immaterial for our purposes, we assume that the forgery has been carried out with due skill, and hence it goes undetected at a visual inspection. However, we assume it is detected with high probability by a suitable machine-learning method based on a local descriptor of the image. More precisely, to classify a generic image X, the detector extracts in sequence the following pieces of information (see Fig.2)

- residual image R, obtained for example, but not necessarily, through

a high-pass linear filter;

- local feature image F, including quantization and coding in the extraction process;
- feature vector $h = \text{hist}(F)$, where hist is the histogram operator;
- estimated class $Y^c \in \{-1, +1\}$, with -1 for pristine and $+1$ for forged image.

Our former assumption on detector reliability translates, therefore, in the detector decisions being with high probability $Y^c(X) = -1$ and $Y^c(X^0) = +1$. The attacker wants to process the forged image so as to a obtain a modified image X^* which evades the detector, while remaining very similar to the original X^0. We consider two alternative scenarios

1. limited knowledge (LK): the attacker knows only the feature extraction process, but does not know the classifier nor the training set used to build it;
2. perfect knowledge (PK): the attacker knows not only the feature extraction procedure but also the classifier or the training set or both.

Depending on which of these scenarios holds, the attacker follows two slightly different strategies.

4.2.1 Limited knowledge

In the LK case, the attacker does not know for sure what the detector will decide for any given input but, lacking any further information, and assuming the detector is generally reliable, works on the reasonable hypothesis that it will classify X as pristine and X^0 as forged. Given this minimal information, a possible strategy is to replace X^0 with a new

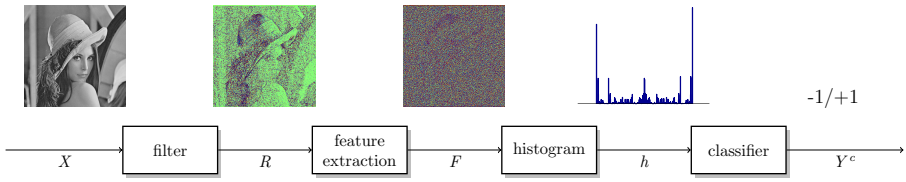


Figure 4.2. Typical workflow of an LD-based machine learning detector. Image X is filtered to obtain its high-pass residual, R, based on which the feature field, F, is computed. Its histogram h is then fed to the classifier.

image X^* such that[1] $h(X^*) = h(X)$, as depicted symbolically in the left part of Fig.3. Among the many modified images respecting this constraint, X^* will be the one most similar to X^0 under a suitable image distortion measure $\phi : \mathscr{X} \times \mathscr{X} \to \overline{R}$. More formally

$$X^* = \arg \min_{Z \in \mathscr{X}} \phi(Z, X^0), \quad \text{s.t. } h(Z) = h(X) \tag{4.1}$$

This formulation presents two major problems. First of all, although there are many feature images with the same histogram, not all of them correspond to valid input images. Depending on the feature extraction process, some combination of values are simply non achievable. Moreover, finding the set of images which respect the constraint of eq.4.1 is unfeasible in practice, as it would require inverting the chain depicted in Fig.2, or else analyzing a huge space of images.

However, since it is reasonable to expect that feature vectors close to $h(X)$ will have themselves a high probability of passing the test (see again Fig.3), we can relax the constraint of eq.4.1, obtaining a more tractable problem

$$Z^* = \arg \min_{Z \in \mathscr{X}} \phi(Z, X^0), \quad \text{s.t. } \psi(h(Z), h(X)) < T_h \tag{4.2}$$

[1] To avoid heavy notation, we use $h(X)$, here, to mean $h(F(X))$

where $\psi : \mathscr{H} \times \mathscr{H} \to \overline{R}$ is a suitable distance defined on the space of histograms $\mathscr{H}$, and T_h a suitable threshold value. Alternatively, we can recast the problem as

$$Z^* = \arg\min_{Z \in \mathscr{X}} \psi(h(Z), h(X)), \quad \text{s.t. } \phi(Z, X^0) < T_X \tag{4.3}$$

switching the roles of image and feature spaces.

In this latter formulation, the constraint is easily satisfied by sampling the image space only in the appropriate region, *e.g.* a ball centered on X^0 if ϕ coincides with the L^2 norm, or an hypercube if the L^∞ norm is preferred. Even so, the solution may still be unsatisfactory, that is, characterized by a large distance in the feature space, and hence not passing the test. Even assuming that X^* is a satisfactory solution, the problem remains of how to achieve it. A large number of algorithms can be used to obtain an approximate solution through a suitable sampling schemes in the search space, all characterized by exceedingly high complexity, however, for a typical-sized image. To limit complexity, we propose an *ad hoc* heuristic, described in subsection 3.4, where sampling is carried out in a space closer to the decision, and hence the search path is more easily steered towards the desired solution.

4.2.2 Perfect knowledge

Now, the attacker can rely on many more pieces of information, as shown symbolically in the right part of Fig.3, where both the decision boundary (that is the classifier) and the labeled training samples are visible. Here we follow the approach of Biggio [100], with the obvious difference that we cannot rely on a gradient descent algorithm. The image is hence modified so that the feature vector travels towards the decision boundary approximately along the orthogonal path. In principle, the updating could stop as soon as the boundary is crossed but this would be fragile w.r.t.

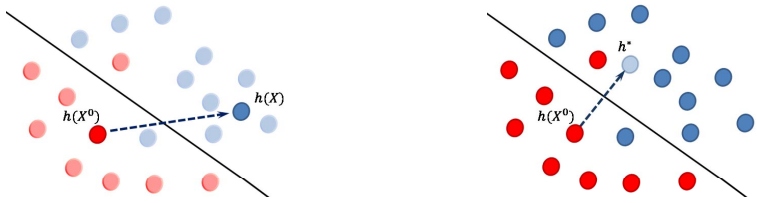


Figure 4.3. Limited knowledge (left) and perfect knowledge (right) strategies: in the first case, the attacker has no information on the detector and aims at the histogram $h(X)$ of the pristine image; in the second case, a suitable synthetic histogram h^* can be built and pursued.

possible changes in the detector. Therefore, we will consider a small safety margin. The obvious advantage w.r.t. the LK case is that the path is typically shorter, implying both a faster convergence and a lower distance from X^0.

If the detector is not known, but the labeled training set is, the algorithm can point towards the closest point corresponding to a pristine image, in which case, however, besides the slower convergence, there is no guarantee to obtain the desired decision, since the training image itself may be mis-classified.

4.2.3 Greedy sampling algorithm

We describe here the sampling algorithm used to find an approximate solution to the problem of eq.4.3. The solution is approximate both because the iterative procedure converges very slowly, calling for some early stopping conditions before reaching the desired feature, and also because its greedy nature implies that it can get trapped in some local minima. In its basic form, the algorithm resembles the Iterated Cconditional Mode (ICM) [128] and its variations.

The algorithm is initialized with the forged image X^0. Then, in the

generic i-th step, X^i is obtained by modifying only one pixel, call it the target t, of the previous image, which is updated so as to maximize locally the objective function. In formulas

$$X^i(p) = \begin{cases} \arg\min_{x(t)} \psi(h(Z), h(X^i)) & p = t \\ X^{i-1}(p) & p \neq t \end{cases} \tag{4.4}$$

This basic step is then repeated until convergence, through some suitable sampling scheme of the image. In particular, it is advisable not to proceed in raster-scan modality to avoid drift effects. Often a pseudo-random scheme is considered, but in our case, given the feature extraction process, we know in advance the footprint of a single-pixel change, so visit the image on a regular grid, suitably large, and slide it by one pixel at each step. Although the value of $\psi(\cdot, \cdot)$ is relatively simple to compute, because of the limited footprint of any change, the overall complexity is still exceedingly high. As a faster alternative, rather than minimizing the distance over $X(t)$, testing all possible values, we can select just one value at random and check whether it reduces the distance, in which case it is accepted. Better yet, when the feature extraction process is relatively simple, we can work in a domain closer to the feature vector, either the residual image or the local feature image, and select in advance only changes that are very likely to reduce the feature distance, with a significant reduction of complexity.

We describe this idea by means of a running example. For the sake of clarity, we select the very simple case of the LBP feature computed over the image (not the residuals) without interpolation. However, with suitable modifications, the same approach can be used in more complex situations. Let us consider the image fragment depicted in Fig.4 (left) in terms of digital numbers. For each point p, the LBP feature is computed

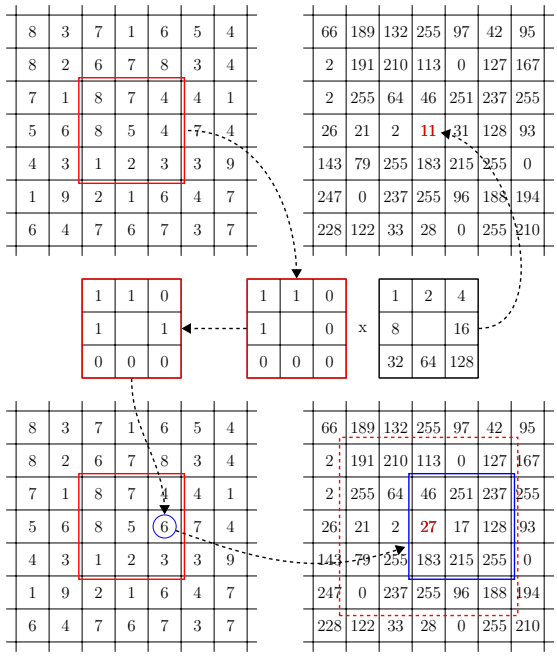


Figure 4.4. A single step of the propose greedy procedure. The target t is the center pixel of the image fragment (top-left). The corresponding LBP string is 11010000, coded as 11 in the feature image (top-right). By increasing the 5-th neighbor of t (bottom-left), the 5-th bit is flipped, changing 11 (undesired) to 27 (desired). In the change footprint (blue, bottom-right): together with the desired switch, there is also a switch from 31 to 17.

as

$$\mathrm{LBP}(p) = \sum_{j=0}^{7} b_j(p) 2^j \tag{4.5}$$

where

$$b_j(p) = \begin{cases} 0 & X(p) \leq X(\eta_j(p)) \\ 1 & \text{otherwise} \end{cases} \tag{4.6}$$

and the $\eta_j(p)$'s are the 8-connected neighbors of p in raster scan order. By applying these formulas on our example image, the string of bits associated

with the target pixel t is $\mathbf{b} = 11010000$ (we drop t from now on, for notational simplicity), corresponding to $F = 11$. When considering pixel t for updating, we inspect the frequency of occurrence of F, namely $h(F)$: if it is already smaller than $h^*(F)$, the desired histogram, no updating takes place. Otherwise, we inspect the features obtained by switching just one bit of the string $\mathbf{b}$, and keep the first one (if any) which reduces the histogram distance. To obtain a switch on the j-th bit, we only have to modify the value of $X(\eta_j(t))$, to change the sign of the difference w.r.t. $X(t)$. Before accepting this change, however, we must check its suitability: it must not increase the distance between X^i and X^0 beyond the threshold, and it must actually reduce the histogram distance, which is not certain because a single pixel alteration impacts on 9 LBP values (its footprint, the blue square in the bottom-right feature image). To simplify this check, we use the L^1 norm for the feature distance. If all controls are passed, the change is accepted. Although still cumbersome, this procedure is much faster than that in the image domain, since most of the selected changes turn out to be acceptable, especially at the beginning.

Needless to say, working on residuals adds further complexity, and in any case the implementation is *ad hoc*, strictly related to the selected feature, and is not granted to work in all cases.

4.3 Experimental results

In this Section we present the results of some preliminary experiments that, although limited in scope, provide much insight into the potential of the proposed method. We consider two LD-based machine learning detectors, the first one using LBP on the original image, called LBP256 in the following, and the second one computing 4-pixel co-occurrences on the residuals of 3rd order liner filter, as proposed in the reference paper [10], and called S3SPAM from now on. Both SVM detectors have been

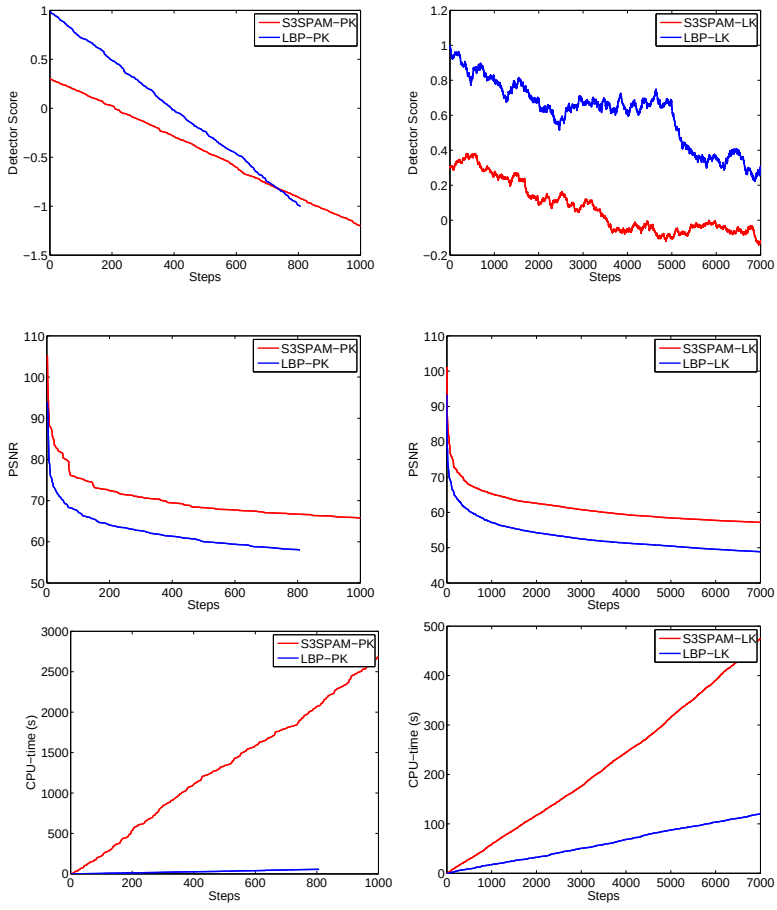


Figure 4.5. Main performance indicators on image 35 of the test set for the PK (left) and LK (right) scenarios.

trained on a dataset of 800 images of size 384×512 pixels, 400 pristine and 400 with random square forgeries (hence, not realistic), of various sizes and processing history. Then, 800 more images of similar characteristics are used as test set. These detectors were tested preliminarily on the dataset of the First Forensic Challenge, providing a score, defined as the

MaxSteps	steps	TPR	PSNR / (min)	Time
500	482	0.89	61.37 / (53.40)	300
1000	888	0.74	59.19 / (52.01)	629
1500	1196	0.47	57.81 / (50.43)	1001
2000	1367	0.23	57.36 / (50.35)	1298
2500	1451	0.11	56.90 / (49.40)	1555
3000	1490	0.05	56.79 / (49.40)	1650
∞	1522	0.00	56.78 / (49.40)	1820

Table 4.1. Performance indicators for the S3SPAM-based detector, PK scenario, averaged over 100 test images.

average probability of correct decision, of 0.86 and 0.91, respectively. On our dataset, results were slightly worse, 0.80 and 0.89, with the detector based on S3SPAM features keeping a clear lead over the simpler LBP-based one. To test our counter-forensic methods we select in advance images that are correctly recognized as forged by both detectors, and with the pristine version recognized as pristine.

Before going to statistical analyses, however, in Fig.5, for one of such images, we plot the main performance indicators as a function of the algorithm progress. More precisely,we report on the x-axis the current step of the algorithm, where each step corresponds to the visit of a target site with possible local updating, and on the y-axis the current detector score (left), image PSNR (middle), and CPU-time (right), for both the perfect knowledge (top row) and limited knowledge (bottom row) scenarios. Consider first the perfect knowledge case. The detector score, related to the distance from the decision boundary, is initially positive, as expected for a forged image, but then drops very quickly (less than 1000 steps) below zero, indicating that the detector is now tricked into classifying the image as pristine. However, before claiming the attack as successful, we must verify that it has left basically unaltered the image, without introducing

visible artifacts. Indeed, the center plot fully supports this case, since a PSNR in the order of 60 dB (for LBP) or 70 dB (for S3SPAM) corresponds to a very high-quality image. Alternative image quality indexes, SSIM, FSIM, not shown here, provide similar indications. Finally,a significant CPU-time is observed for the S3SPAM-based detector, due to the need to re-classify the feature each time to compute the score and, not last, to our current implementation in Matlab, certainly inefficient.

In the limited-knowledge scenario things are much different. The score does not decrease monotonically anymore, because moves going towards $h(X)$ may occasionally increase the distance from the decision boundary. In general, the convergence is much slower, and the score takes more than 3000 steps to become negative with the S3SPAM feature, and it never does with LBP (the attack fails). These results may look surprising. With reference to Fig.3, they could be justified only if the line from $h(X^0)$ to $h(X)$ traveled almost parallel to the decision hyperplane. In fact, this is exactly the case: with features living in a space with hundreds of dimensions, the line orthogonal to the boundary can explain only a tiny fraction of the distance between two points, and hence most of the updates are just useless. With so many updating steps, also the PSNR decreases, but remains always above a safe 50 dB limit. As a positive side, complexity decreases significantly w.r.t. the PK case.

These remarks are confirmed by experiments on the full dataset. In particular, the counter-forensic attack has been applied to 100 images chosen as said before. Results are reported in Tables I to IV. We draw attention only on a few remarkable results: first of all, in the perfect knowledge scenario, the attack has always success. The long time required for the S3SPAM-based detector can be probably cut by 90% with a careful implementation in a compiled language. Instead, in the limited knowledge scenario, results are much worse, and the CPU time even larger. In all cases, the PSNR remains pretty large, and in no instance goes below 40dB,

MaxSteps	steps	TPR	PSNR / (min)	Time
500	140	0.80	57.60 / (49.60)	25
1000	165	0.40	54.10 / (46.60)	50
1500	173	0.20	51.70 / (45.00)	78
2000	179	0.01	49.90 / (43.70)	107
2500	184	0.01	48.70 / (42.70)	137
3000	185	0.00	47.90 / (41.90)	164
∞	185	0.00	47.90 / (41.90)	164

Table 4.2. Performance indicators for the LBP-based detector, PK scenario, averaged over 100 test images.

MaxSteps	steps	TPR	PSNR / (min)	Time
1000	992	0.92	61.47 / (55.56)	945
5000	4127	0.85	56.01 / (49.24)	1515
10000	7877	0.83	53.87 / (46.37)	1637
30000	22168	0.75	50.70 / (42.65)	2311
50000	31010	0.50	49.71 / (40.84)	3000
100000	37588	0.33	49.07 / (40.24)	3842

Table 4.3. Performance indicators for the S3SPAM-based detector, LK scenario, averaged over 100 test images.

ensuring that the attack is not perceivable by visual inspection. To further stress this point, we conclude by showing one of the images of Fig.1 after our attack, both in the PK and LK scenarios. The output images appear as identical, and identical to the original forged image. Although there is much room for further research, and better algorithms can be certainly singled out, our techniques provide already encouraging and sometimes quite good results, attacking successfully some of the best image forgery detectors proposed in recent years.

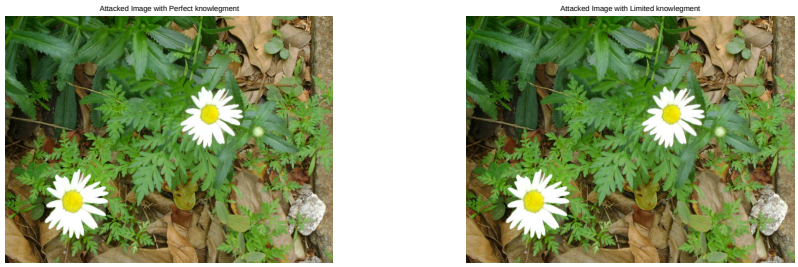


Figure 4.6. Output image after counter-forensic attacks in the PK (left) and LK (right) scenarios.

10.1007/978-3-319-68548-9

Conclusions

This thesis deals with image source identification. In particular, two different type of problems have been faced: PRNU-based blind image clustering and camera model classification. In both cases special attention has been devoted to JPEG-compressed and resized images, so as to mimick the processing operations that images undergo when posted on social networks.

For what concern the PRNU-based approach, after computing the image noise residuals, correlation clustering and consensus clustering are used to obtain a first conservative data partition, ideally free of false positives. Then, an *ad hoc* refinement algorithm is used to obtain the final clustering by alternating PRNU estimate improvement and cluster merging. Results on several datasets extracted from the Dresden database, both pristine and resized/compressed, prove the proposed method to outperform the-state-of-the-art and guarantee higher robustness to image quality impairments. A remarkable feature of the proposed method is that no user intervention is required, to provide parameters or external training sets. In addition, by a judicious choice of clustering and estimation tools, the computational complexity is always quite limited.

Then, we used the proposed algorithm to face forgery localization in a blind scenario. After images have been clustered, PRNU is estimated

and used to tell apart pristine images from forged ones by performing a pixel-level analysis. Clearly, each step can be a possible source of error and experimental analysis highlighted how performance degraded because of the errors introduced at each single step. It turned out that for the original images the performance of all clustering algorithms are high enough to create clusters with a low false positive rate. This allows the forgery detector to perform as well as in the ideal case. However, in the JPEG-compressed dataset, a performance drop can be observed since the clustering becomes less accurate and more fragmented. Further experiments are certainly needed for a full assessment of the performance. In particular, since practical applications often deal with very large datasets (from thousands to millions of images), a study of the algorithm behavior in the presence of PRNU compression [129] or fast method [130] seems necessary.

For the camera model identification problem, the use of co-occurrence based features have been analyzed. Here a different level of knowledge on the training set and pre-processing on images is considered. Then a deep-learning approach is shown to outperform all state-of-the-art methods, giving promising results even with small patches.

Finally, the last chapter of this thesis is devoted to machine learning-based counter-forensics and show that a malicious attacker, aware of the specific features used by the system, can easily fool the detector. This raises up security issues that should be addressed by the researchers in the next years to provide more robust approaches.

In general, there are several open questions for future research. A first issue is how to merge the PRNU traces with the camera model ones to improve performance even in terms of speed and accuracy. The extension to videos is also not trivial, not only for the different nature of the data, but also for the largely increased computational load. Another interesting scenario arises if an expert attacker performs a counter-forensics approach to fool the deep learning paradigm [131]. All such problems strongly stim-

ulate research and make multimedia forensics one of the most attractive topic in the signal processing field.

Bibliography

[1] M. Kirchner and T. Gloe. Forensic camera model identification. In *Handbook of Digital Forensics of Multimedia Data and Devices*. Wiley-IEEE Press, 2015.

[2] J. Lukàš, J. Fridrich, and M. Goljan. Digital camera identification from sensor pattern noise. *IEEE Transactions on Information Forensics and Security*, 1(2):205–214, june 2006.

[3] T. Filler, J. Fridrich, and M. Goljan. Using sensor pattern noise for camera model identification. In *IEEE International Conference on Image Processing*, pages 1296–1299, 2008.

[4] M. Chen, J. Fridrich, M. Goljan, and J. Lukás. Determining image origin and integrity using sensor noise. *IEEE Transactions on Information Forensics and Security*, 3(1):74–90, March 2008.

[5] R. Böhme and M. Kirchner. Counter-forensics: attacking image forensics. In *Digital Image Forensics (Chapter 10)*. Springer, New York, 2012.

[6] Y. Huang, J. Zhang, and Heyan Huang. Camera model identification with unknown models. *IEEE Transactions on Information Forensics and Security*, 10(12):2692–2704, 2015.

[7] F.O. Costa, E. Silva, M. Eckmann, W.J. Scheirer, and A. Rocha. Open set source camera attribution and device linking. *Pattern Recognition Letters*, 39:92–101, 2014.

[8] G. J. Bloy. Blind camera fingerprinting and image clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(3):532–534, March 2008.

[9] D. Cozzolino, D. Gragnaniello, and L.Verdoliva. Image forgery detection through residual-based local descriptors and block-matching. In *IEEE International Conference on Image Processing (ICIP)*, pages 5297–5301, October 2014.

[10] J. Fridrich and J. Kodovsky. Rich models for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security*, 7:868–882, 2012.

[11] A. Tuama, F. Comby, and M. Chaumont. Source camera model identification using features from contaminated sensor noise. In *International Workshop on Digital-forensics and Watermarking*, October 2015.

[12] F. Marra, G. Poggi, C. Sansone, and L. Verdoliva. Evaluation of residual-based local features for camera model identification. In *New Trends in Image Analysis and Processing – ICIAP 2015 Workshops*, pages 11–18, 2015.

[13] G. Chierchia, S. Parrilli, G. Poggi, C. Sansone, and L. Verdoliva. On the influence of denoising in PRNU based forgery detection. In

2nd ACM workshop on Multimedia in Forensics, Security and Intelligence, pages 117–122, 2010.

[14] A. Tuama, F. Comby, and M. Chaumont. Camera model identification with the use of deep convolutional neural networks. In *IEEE International Workshop on Information Forensics and Security*, pages 1–6, 2016.

[15] L. Bondi, L. Baroffio, D. Guera, P. Bestagini, E. J. Delp, and S. Tubaro. First steps toward camera model identification with convolutional neural networks. *IEEE Signal Processing Letters*, 24(3):259–263, March 2017.

[16] F. Marra, G. Poggi, C. Sansone, and L. Verdoliva. A study of co-occurrence based local features for camera model identification. *Multimedia Tools and Applcations*, pages 1–17, 2017.

[17] F. Marra, G. Poggi, C. Sansone, and L. Verdoliva. A deep learning approach for iris sensor model identification. *Pattern Recognition Letters*, 2017 (in press).

[18] A.L. Sandoval Orozco, J.R. Corripioand L.J. Garcíia Villalba, and J.C.H. Castro. Image source acquisition identification of mobile devices based on the use of features. *Multimedia Tools and Applications*, in press 2015.

[19] S. Bayram, H.T. Sencar, N. Memon, and I. Avcibas. Source camera identification based on CFA interpolation. In *IEEE Int. Conference on Image Processing*, pages 69–72, 2005.

[20] A.C. Popescu and H. Farid. Exposing digital forgeries by detecting traces of resampling. *IEEE Transactions on Signal Processing*, 53(2):758–767, 2005.

[21] A. Swaminathan, M. Wu, and K. J. Ray Liu. Rich models for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security*, 2(1):91–105, 2007.

[22] H. Cao and A.C. Kot. Accurate detection of demosaicing regularity for digital image forensics. *IEEE Trans. on Information Forensics and Security*, 4(4):899–910, 2009.

[23] S. Bayram, H.T. Sencar, and N. Memon. Improvements on source camera-model identification based on CFA. In *Advances in Digital Forensics II, IFIP International Conference on Digital Forensics*, pages 289–299, 2006.

[24] N. Fan, C. Jin, and Y. Huang. Source Camera Identification by JPEG Compression Statistics for Image Forensics. In *TENCON*, pages 1–4, 2006.

[25] T.H. Thai, F. Retraint, and R. Cogranne. Camera model identification based on DCT coefficient statistics. *Digital Signal Processing*, 4:88–100, 2015.

[26] L.T. Van, S. Emmanuel, and M.S. Kankanhalli. Identifying source cell phone using chromatic aberration. In *IEEE International Conference on Multimedia and Expo*, pages 883–886, 2007.

[27] T.H. Thai, R. Cogranne, and F. Retraint. Camera model identification based on the heteroscedastic noise model. *IEEE Transactions on Image Processing*, 23(1):250–263, 2014.

[28] M. Kharrazi, H.T. Sencar, and N. Memon. Blind source camera identification. In *IEEE International Conference on Image Processing*, pages 709–712, 2004.

[29] I. Avcibaş, N. Memon, and B. Sankur. Steganalysis using image quality metrics. *IEEE Transactions on Image Processing*, 12(2):221–229, February 2003.

[30] K. Goyal, R. Panwar, and N. Khanna. Evaluation of IQM's effectiveness for cell phone identification using captured videos and images. In *International Conference on Power, Control and Embedded Systems*, pages 1–6, 2014.

[31] O. Çeliktutan, B. Sankur, and I. Avcibaş. Blind identification of source cell-phone model. *IEEE Transactions on Information Forensics and Security*, 3(3):553–566, 2008.

[32] T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002.

[33] S. Lyu and H. Farid. Steganalysis using higher-order image statistics. *IEEE Transactions on Information Forensics and Security*, 1(1):111–119, January 2006.

[34] T. Gloe. Feature-based forensic camera model identification. In *LNCS Transactions on Data Hiding and Multimedia Security VIII*, volume 7228, pages 42–62, 2012.

[35] G. Xu and Y.Q. Shi. Camera model identification using local binary patterns. In *IEEE International Conference on Multimedia and Expo*, pages 392–397, 2012.

[36] F. Razzazi and A. Seyedabadi. A robust feature for single image camera identification using local binary patterns. In *IEEE International Symposium on Signal Processing and Information Technology*, pages 462–467, 2014.

[37] D. Gragnaniello, G. Poggi, C. Sansone, and L. Verdoliva. An investigation of local descriptors for biometric spoofing detection. *IEEE Transactions on Information Forensics and Security*, 10(4):849–863, 2015.

[38] G. Xu, S. Gaon, Y.Q. Shi, R.M. Hu, and W. Su. Camera-model identification using markovian transition probability matrix. In *Digital Watermarking, LNCS*, volume 5703, pages 294–307, 2009.

[39] M. Kirchner and J. Fridrich. On detection of median filtering in images. In *SPIE, Electronic Imaging, Media Forensics and Security XII*, pages 101–112, 2010.

[40] L. Verdoliva, D. Cozzolino, and G. Poggi. A feature-based approach for image tampering detection and localization. In *IEEE International Workshop on Information Forensics and Security*, pages 149–154, 2014.

[41] M. Goljan, J. Fridrich, and R. Cogranne. Rich model for steganalysis of color images. In *2014 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 185–190, Dec 2014.

[42] M. Goljan and J. Fridrich. CFA-aware features for steganalysis of color images. In *SPIE, Electronic Imaging, Media Watermarking, Security and Forensics*, volume 9409, 2015.

[43] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Conference on Neural Information Processing Systems*, page 1097–1105, 2012.

[44] T. Gloe and R. Böhme. The Dresden image database for benchmarking digital image forensics. *Journal of Digital Forensic Practice*, 3(2-4):150–159, 2010.

[45] N. Kohli, D. Yadav, M. Vatsa, and R. Singh. Revisiting iris recognition with color cosmetic contact lenses. In *International Conference on Biometrics (ICB)*, pages 1–7, June 2013.

[46] D. Yadav, N. Kohli, J.S. Doyle, R. Singh, M. Vatsa, and K.W. Bowyer. Unraveling the effect of textured contact lenses on iris recognition. *IEEE Transactions on Information Forensics and Security*, 9(5):851–862, May 2014.

[47] J. S. Doyle, K. W. Bowyer, and P. J. Flynn. Variation in accuracy of textured contact lens detection based on sensor and lens pattern. In *IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS)*, pages 1–7, Sept 2013.

[48] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. John Wiley & Sons, New York, 2 edition, 2001.

[49] S.S. Arora, M. Vatsa, R. Singh, and A. Jain. On iris camera interoperability. In *International Workshop on Biometrics and Forensics*, pages 346–352, 2012.

[50] F. Bertini, R. Sharma, A. Ianni, and D. Montesi. Profile resolution across multilayer networks through smartphone camera fingerprint. In *19th International Database Engineering and Applications Symposium*, pages 23–32, 2015.

[51] R. Satta and A. Ciardulli. Sensor pattern noise and image similarity for picture-to-identity linking. *IET Computer Vision*, pages 1–12, 2015.

[52] F.M. Naini, J. Unnikrishnan, P. Thiran, and M. Vetterli. Where you are is who you are: User identification by matching statistics. *IEEE Transactions on Information Forensics and Security*, 11(2):358–372, Feb 2016.

[53] A. L. N. Fred and A. K. Jain. Combining multiple clusterings using evidence accumulation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(6):835–850, June 2005.

[54] S. Vega-Pons and J. Ruiz-Shulcloper. A survey of clustering ensemble algorithms. *International Journal of Pattern Recognition and Artificial Intelligence*, 25(03):337–372, 2011.

[55] F. Marra, G. Poggi, C. Sansone, and L. Verdoliva. Correlation clustering for prnu-based blind image source identification. In *IEEE International Workshop on Information Forensics and Security*, pages 1–6, 2016.

[56] F. Marra, G. Poggi, C. Sansone, and L. Verdoliva. Blind prnu-based image clustering for source identification. *IEEE Transactions on Information Forensics and Security*, 12(9):2197–2211, Sept 2017.

[57] W. H. Equitz. A new vector quantization clustering algorithm. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(10):1568–1575, Oct 1989.

[58] C. T. Li. Unsupervised classification of digital images using enhanced sensor pattern noise. In *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, pages 3429–3432, May 2010.

[59] C.-T. Li. Source camera identification using enhanced sensor pattern noise. *IEEE Transactions on Information Forensics and Security*, 5(2):280–287, 2010.

[60] R. Caldelli, I. Amerini, F. Picchioni, and M. Innocenti. Fast image clustering of unknown source images. In *IEEE International Workshop on Information Forensics and Security*, pages 1–5, 2010.

[61] S. Luan, X. Kong, B. Wang, Y. Guo, and X. You. Silhouette coefficient based approach on cell-phone classification for unknown source images. In *IEEE International Conference on Communications*, pages 6744–6747, 2012.

[62] L. J. García Villalba, A. L. Sandoval Orozco, and J. R. Corripio. Smartphone image clustering. *Expert Syst. Appl.*, 42(4):1927–1940, March 2015.

[63] F. Gisolf, P. Barens, E. Snel, A. Malgoezar, M. Vos, A. Mieremet, and Z. Geradts. Common source identification of images in large databases. *Forensic Science International*, 244:222–230, 2014.

[64] O.M. Fahmy. An efficient clustering technique for cameras identification using sensor pattern noise. In *International Conference on Systems, Signals and Image Processing*, pages 249–252, 2015.

[65] B.-B. Liu, H.-K. Lee, Y. Hu, and C.-H. Choi. On classification of source cameras: a graph based approach. In *IEEE International Workshop on Information Forensics and Security*, pages 1–5, 2010.

[66] S.X. Yu and J. Shi. Multiclass spectral clustering. In *IEEE International Conference on Computer Vision*, pages 313–319, 2003.

[67] I. Amerini, R. Caldelli, P. Crescenzi, A. Del Mastio, and A. Marino. Blind image clustering based on the normalized cuts criterion for camera identification. *Signal Processing: Image Communication*, 29(8):831 – 843, 2014.

[68] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, August 2000.

[69] Dong Huang, Jian-Huang Lai, and Chang-Dong Wang. Combining multiple clusterings via crowd agreement estimation and multi-

granularity link analysis. *Neurocomput.*, 170(C):240–250, December 2015.

[70] G.E. Healey and R. Kondepudy. Radiometric ccd camera calibration and noise estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(3):267–276, March 1994.

[71] N. Bartlow, N. Kalka, B. Cukic, and A. Ross. Identifying sensors from fingerprint images. In *IEEE International Conference on Computer Vision and Pattern Recognition Workshops*, pages 78–84, 2009.

[72] G.M. Farinella, M.V. Giuffrida, V. Digiacomo, and S. Battiato. On blind source camera identification. In *proc. of Advanced Concepts for Intelligent Vision Systems*, pages 464–463, October 2015.

[73] D. Cozzolino, D. Gragnaniello, and L.Verdoliva. Image forgery localization through the fusion of camera-based, feature-based and pixel-based techniques. In *IEEE International Conference on Image Processing (ICIP)*, pages 5237–5241, October 2014.

[74] G. Chierchia, G. Poggi, C. Sansone, and L. Verdoliva. A Bayesian-MRF approach for PRNU-based image forgery detection. *IEEE Transactions on Information Forensics and Security*, 9(4):554–567, April 2014.

[75] G. Chierchia, D. Cozzolino, G. Poggi, C. Sansone, and L. Verdoliva. Guided filtering for PRNU-based localization of small-size image forgeries. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6231–6235, May 2014.

[76] X. Lin and C. T. Li. Preprocessing reference sensor pattern noise via spectrum equalization. *IEEE Transactions on Information Forensics and Security*, 11(1):126–140, Jan 2016.

[77] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. In *Foundations of Computer Science, 2002. Proceedings. The 43rd Annual IEEE Symposium on*, pages 238–247, 2002.

[78] S. Bagon and M. Galun. Large scale correlation clustering optimization. *CoRR*, abs/1112.2903, 2011.

[79] J. Wu, H. Liu, H. Xiong, J. Cao, and J. Chen. K-means-based consensus clustering: A unified view. *IEEE Transactions on Knowledge and Data Engineering*, 27(1):155–169, 2015.

[80] A. Strehl and J. Ghosh. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.*, 3:583–617, March 2003.

[81] X. Z. Fern and C. E. Brodley. Solving cluster ensemble problems by bipartite graph partitioning. In *Proceedings of the Twenty-first International Conference on Machine Learning*, pages 36–, 2004.

[82] M. Yousefnezhad and D. Zhang. Weighted spectral cluster ensemble. *15th Int. Conf. Data Mining*, abs/1604.07178:549–559, Sept 2015.

[83] E. F. Lock and D. B. Dunson. Bayesian consensus clustering. *Bioinformatics*, 29(20):2610–2616, 2013.

[84] A. K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recogn. Lett.*, 31(8):651–666, June 2010.

[85] James Surowiecki. *The Wisdom of Crowds*. Anchor, 2005.

[86] Y. Şenbabaoğlu, G.e Michailidis, and J. Z. Li. Critical limitations of consensus clustering in class discovery. *Scientific Reports*, 4, 8 2014.

[87] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, Aug 2007.

[88] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer. Optimizing binary MRFs via extended roof duality. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007.

[89] M. Moltisanti, A. Paratore, S. Battiato, and L. Saravo. *Image Manipulation on Facebook for Forensics Evidence*, pages 506–517. 2015.

[90] W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[91] L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985.

[92] T. Gloe, S. Pfennig, and M. Kirchner. Unexpected artefacts in PRNU based camera identification: A 'dresden image database' case-study. In *14th ACM Workshop on Multimedia Security*, pages 109–114, 2012.

[93] G.Chierchia, S.Parrilli, G.Poggi, C.Sansone, and L.Verdoliva. PRNU-based detection of small size image forgeries. In *International Conference on Digital Signal Processing*, pages 1–6, July 2011.

[94] S. Chakraborty and M. Kirchner. PRNU-based forgery detection with discriminative random fields. In *International Symposium on Electronic Imaging: Media Watermarking, Security, and Forensics*, February 2017.

[95] P. Korus and J. Huang. Multi-scale analysis strategies in PRNU-based tampering localization. *IEEE Transactions on Information Forensics and Security*, 12(4):809–824, April 2017.

[96] D. Cozzolino, F. Marra, G. Poggi, C. Sansone, and L. Verdoliva. Prnu-based forgery localization in a blind scenario. In *International Conference on Image Analysis and Processing (ICIAP)*, pages 569–579, 2017.

[97] M. Goljan. Digital camera identification from images – estimating false acceptance probability. In *Proc. 8th Int. Workshop Digital Watermarking*, 2008.

[98] M. Goljan, J. Fridrich, and T. Filler. Large scale test of sensor fingerprint camera identification. In *SPIE, Electronic Imaging, Media Forensics and Security*, volume 7254, pages 72540I–72540I–12, 2009.

[99] F. Marra, G. Poggi, F. R., C. Sansone, and L. Verdoliva. Counter-forensics in machine learning based forgery detection. In *SPIE, Electronic Imaging, Media Watermarking, Security, and Forensics*, volume 9409, 2015.

[100] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Srndić, P. Laskov, G. Giacinto, and F. Roli. Evasion attacks against machine learning at test time. In *European Conference on Machine Learning and Knowledge Discovery in Databases*, volume 8190, pages 387–402, 2013.

[101] M. Kirchner and R. Böhme. Tamper hiding: Defeating image forensics. In *International Conference on Information Hiding*, pages 326–341, 2007.

[102] M.C. Stamm and K.J. Ray Liu. Anti-forensics of digital image compression. *IEEE Transactions on Information Forensics and Security*, 6(3):1050–1065, September 2011.

[103] G. Valenzise, M. Tagliasacchi, and S. Tubaro. Revealing the traces of jpeg compression anti-forensics. *IEEE Transactions on Information Forensics and Security*, 8(2):335–349, february 2013.

[104] T. Gloe, M. Kirchner, A. Winkler, and R. Böhme. Can we trust digital image forensics? In *ACM 15th Int. Conf. Multimedia*, pages 78–86, 2007.

[105] M. Goljan, J. Fridrich, and M. Chen. Defending against fingerprint-copy attack in sensor-based camera identification. *IEEE Transactions on Information Forensics and Security*, 6(1):227–236, march 2011.

[106] F. Marra, F. Roli, D. Cozzolino, C. Sansone, and L. Verdoliva. Attacking the triangle test in sensor-based camera identification. In *IEEE International Conference on Image Processing*, pages 5307–5311, Oct 2014.

[107] M.C. Stamm, S. Lin, and K.J.R. Liu. Forensics vs. anti-forensics: A decision and game theoretic framework. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1749–1752, 2012.

[108] M. Barni and B. Tondi. The source identification game: An information-theoretic perspective. *IEEE Transactions on Information Forensics and Security*, 8(3):450–463, March 2013.

[109] M. Barni and F. Pérez-González. Coping with the enemy: Advances in adversary-aware signal processing. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8682–8686, May 2013.

[110] M. Barni, M. Fontani, and B. Tondi. A universal technique to hide traces of histogram-based image manipulations. In *ACM Multimedia and Security Workshop*, pages 97–104, 2012.

[111] P. Comesaña-Alfaro and F. Pérez-González. Optimal counterforensics for histogram-based forensics. In *IEEE International Conference*

on Acoustics, Speech and Signal Processing, pages 3048–3052, May 2013.

[112] M. Iuliani, S. Rossetto, T. Bianchi, A. De Rosa, A. Piva, and M. Barni. Image counter-forensics based on feature injection. In *SPIE, Electronic Imaging, Media Watermarking, Security, and Forensics*, volume 9028, pages 1–15, 2014.

[113] P. Comesana and F. Pèrez-Gonzàlez. The optimal attack to histogram-based forensic detectors is simple(x). In *IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 137–142, Dec 2014.

[114] Y.Q. Shi, C. Chen, and G. Xuan. Steganalysis versus splicing detection. In *International Workshop on Digital Watermarking*, volume 5041, pages 158–172, 2008.

[115] Z. He, W. Lu, W. Sun, and J. Huang. Digital image splicing detection based on markov features in dct and dwt domain. *Pattern Recognition*, 45:4292–4299, 2012.

[116] X. Zhao, S. Wang, S. Li, J. Li, and Q. Yuan. Image splicing detection based on noncausal markov model. In *IEEE International Conference on Image Processing*, pages 4462–4466, 2013.

[117] D. Cozzolino, D. Gragnaniello, and L. Verdoliva. Image forgery detection through residual-based local descriptors and block-matching. In *IEEE International Conference on Image Processing (ICIP)*, pages 5297–5301, 2014.

[118] D. Cozzolino, G. Poggi, and L. Verdoliva. Splicebuster: A new blind image splicing detector. In *IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–6, 2015.

[119] D. Cozzolino and L. Verdoliva. Single-image splicing localization through autoencoder-based anomaly detection. In *IEEE International Workshop on Information Forensics and Security*, pages 1–6, Dec 2016.

[120] C. Cachin. An information theoretic model for steganography. In *Information and Computation*, volume 1525, pages 306–318, 1998.

[121] K. Solanki, K. Sullivan, U. Madhow, B.S. Manjunath, and S. Chandrasekaran. Statistical restoration for robust and secure steganography. In *IEEE International Conference on Image Processing*, pages 1118–1121, 2005.

[122] A. Sarkar, K. Solanki, U. Macdhow, S. Chandrasekaran, and B.S. Manjunath. Secure steganography: statistical restoration of the second order dependencies for improved security. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 277–280, 2007.

[123] L. Chen, S. Wang, S. Li, and J. Li. Countering universal image tampering detection with histogram restoration. In *Proc. of the 11th international Conference on Digital Forensics and Watermaking*, pages 282–289, 2012.

[124] M. Barreno, B. Nelson, R. Sears, A.D. Joseph, and J.D. Tygar. Can machine learning be secure? In *ACM Symp. on Information, Computer and Comm. Sec.*, pages 16–25, 2006.

[125] L. Huang, A.D. Joseph, B. Nelson, B. Rubinstein, and J.D. Tygar. Adversarial machine learning. In *ACM Workshop on Artificial Intell. and Security*, pages 43–57, 2011.

[126] A. Melloni, P. Bestagini, A. Costanzo, M. Barni, M. Tagliasacchi, and S. Tubaro. Attacking image classification based on bag-of-visual-

words. In *IEEE International Workshop on Information Forensics and Security*, pages 103–108, 2013.

[127] A. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *IEEE International Conference on Computer Vision and Pattern Recognition*, 2015.

[128] J.E. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society*, 48(3):259–302, 1986.

[129] D. Valsesia, G. Colucci, T. Bianchi, and E. Magli. Compressed fingerprint matching and camera identification via random projections. *IEEE Transactions on Information Forensics and Security*, 10(7):1472–1485, 2015.

[130] F. Pérez-González, M. Masciopinto, I. González-Iglesias, and P. Comesaña. Fast sequential forensic detection of camera fingerprint. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3902–3906, Sept 2016.

[131] D. Cozzolino, G. Poggi, and L. Verdoliva. Recasting residual-based local descriptors as convolutional neural networks: An application to image forgery detection. In *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security*, pages 159–164, 2017.